User Stories : AI FinanceMaster App

Testing and Quality Assurance

This epic covers testing strategies including unit, integration, E2E, performance, security, and accessibility testing to ensure product quality and compliance.

6 stories

Conduct Accessibility Testing for WCAG 2.1 AA Compliance

medium

As a UX designer, I want to ensure the app meets WCAG 2.1 AA accessibility standards, so that it is usable by people with disabilities.

Dependencies:

c2295a06... b182b181...

Acceptance Criteria:

- ✓ Functional: App passes automated accessibility checks using axe-core.
- ✓ Functional: Manual accessibility testing covers keyboard navigation and screen readers.
- ✓ Technical: Accessibility issues are tracked and fixed.
- ✓ Functional: Accessibility compliance is documented.
- ✓ Technical: Accessibility testing integrated into CI pipeline.

Story Points: 3

Estimated Effort: 5 hours

Perform Security Testing and Vulnerability Scanning

high

As a security engineer, I want to perform security testing including penetration tests and vulnerability scans, so that the platform is secure and compliant.

Dependencies:

ef196ec1... 42e97089...

Acceptance Criteria:

- ✓ Functional: OWASP ZAP scans are performed regularly.
- ✓ Functional: Penetration tests cover critical components and APIs.
- ✓ Technical: Vulnerabilities are tracked and remediated promptly.
- ✓ Functional: Security testing results are documented for compliance.
- ✓ Technical: Automated security scans run in CI pipeline.

Story Points: 5

Estimated Effort: 8 hours

Conduct Performance Testing for Scalability and Latency

high

As a performance engineer, I want to conduct load and stress testing, so that the system meets performance and scalability requirements.

Dependencies:

9c9f6ceb... 46e3abf9...

Acceptance Criteria:

- ✓ Functional: System supports 10,000 concurrent users with <1s response time.
- ✓ Technical: Load tests simulate realistic user behavior and data volumes.
- ✓ Functional: Bottlenecks and failures are identified and documented.
- ✓ Technical: Performance improvements are tracked over time.
- ✓ Functional: Reports are generated for stakeholders.

Story Points: 5

Estimated Effort: 8 hours

Implement End-to-End (E2E) Tests for Critical User Flows

high

As a QA engineer, I want to write E2E tests for critical user flows like onboarding, account linking, and subscription management, so that user experience is validated.

Dependencies:

263ada52... d3dd22b8... ed16214a...

Acceptance Criteria:

- ✓ Functional: E2E tests cover onboarding, authentication, subscription management, and alerts.
- ✓ Technical: Tests simulate user interactions in browser.
- ✓ Functional: Tests run in CI pipeline with pass/fail reports.
- ✓ Technical: Tests handle asynchronous operations and API delays.
- ✓ Functional: Failures trigger alerts to QA team.

Story Points: 8

Estimated Effort: 13 hours

Implement Integration Tests for API Endpoints

high

As a QA engineer, I want to write integration tests for API endpoints, so that end-to-end functionality is verified across services.

Dependencies:

263ada52... d3dd22b8...

Acceptance Criteria:

- ✓ Functional: Integration tests cover all critical API endpoints.
- ✓ Technical: Tests use real or mocked databases and services.

- Functional: Tests validate data flows and error scenarios.
- Technical: Tests run in CI pipeline with reports.
- Functional: Failures are reported and tracked.

Story Points: 5

Estimated Effort: 8 hours

Implement Unit Tests for Backend Services

high

As a developer, I want to write unit tests for backend services, so that individual components function correctly and regressions are prevented.

Dependencies:

263ada52...

Acceptance Criteria:

- Functional: Unit tests cover at least 80% of backend code.
- Technical: Tests run automatically in CI pipeline.
- Functional: Tests validate business logic and error handling.
- Technical: Mock external dependencies for isolation.
- Functional: Test failures block merges.

Story Points: 5

Estimated Effort: 8 hours

Initial Project Setup and Infrastructure

This epic covers the critical foundation work including repository initialization, development environment setup, CI/CD pipeline, database migration framework, testing framework, code quality tools, and documentation framework.

7 stories

Spike: Documentation Framework Setup

medium

As a product manager, I want to set up documentation frameworks, so that API docs, user guides, and technical docs are maintained effectively.

Acceptance Criteria:

- API docs are accessible and versioned
- Documentation is kept up-to-date with code
- Team is trained on documentation process

Story Points: 3

Spike: Code Quality Tools Setup

medium

As a developer, I want to set up code quality tools like ESLint and Prettier, so that code style is consistent and maintainable.

Acceptance Criteria:

- ✓ Code fails CI if lint errors exist
- ✓ Developers can auto-format code
- ✓ Code style violations are minimized

Story Points: 2

Spike: Testing Framework Setup

high

As a QA engineer, I need to set up testing frameworks for unit, integration, and E2E tests, so that code quality and functionality are verified automatically.

Acceptance Criteria:

- ✓ Tests run automatically in CI pipeline
- ✓ Tests cover basic app functionality
- ✓ Test failures block merges

Story Points: 5

Spike: Database Migration Framework Setup

high

As a backend engineer, I need to set up a database migration framework, so that schema changes can be managed and deployed safely.

Acceptance Criteria:

- ✓ Migrations can be applied and rolled back in dev and staging
- ✓ Migration history is tracked in database
- ✓ Migration failures are handled gracefully

Story Points: 3

Spike: CI/CD Pipeline Setup with GitHub Actions

high

As a DevOps engineer, I need to set up automated CI/CD pipelines, so that code is built, tested, and deployed reliably with minimal manual intervention.

Acceptance Criteria:

- ✓ Pipeline triggers on pull requests and merges
- ✓ Build artifacts are stored and versioned
- ✓ Deployments use blue-green strategy

Story Points: 5

Spike: Development Environment Setup with Docker

high

As a developer, I need to set up a consistent local development environment using Docker, so that all team members can develop and test reliably.

Acceptance Criteria:

- ✓ Developers can start full stack environment with one command
- ✓ Environment matches production dependencies and versions
- ✓ Hot reload supported for frontend and backend

Story Points: 3

Spike: Repository Initialization and Project Scaffolding

high

As a developer, I need to set up the initial code repository and project scaffolding, so that development can start with a clean and organized codebase.

Acceptance Criteria:

- ✓ Repository is accessible to all team members
- ✓ Project scaffolding supports React frontend and Node.js backend
- ✓ Basic build scripts are functional

Story Points: 3

Data Security and Privacy

This epic focuses on protecting sensitive financial and personal data from unauthorized access and breaches, including encryption, access controls, audit logging, and security protocol updates.

3 stories

Regularly Update Security Protocols and Incident Response

high

As the security team, I want to regularly update security protocols and maintain an incident response plan, so that the platform remains secure against emerging threats.

Dependencies:

ef196ec1...

Acceptance Criteria:

- ✓ Functional: Security protocols are reviewed and updated quarterly.
- ✓ Functional: Incident response plan is documented and tested regularly.
- ✓ Technical: Security patches and updates are applied promptly.
- ✓ Functional: Security incidents are logged, reported, and resolved.
- ✓ Technical: Penetration testing is performed annually.

Story Points: 3

Estimated Effort: 5 hours

Implement Access Controls and Audit Logging

high

As the system, I want to enforce strict access controls and maintain audit logs of all sensitive data access and actions, so that security and compliance are ensured.

Dependencies:

defa1f08... c3bf455c...

Acceptance Criteria:

- ✓ Technical: Role-based access control is enforced at all service and data layers.
- ✓ Functional: All access to sensitive data is logged with user and timestamp.
- ✓ Technical: Audit logs are immutable and stored securely.
- ✓ Functional: Logs are accessible for compliance audits and incident investigations.
- ✓ Technical: Access violations trigger alerts and are investigated.

Story Points: 5

Estimated Effort: 8 hours

Implement Encryption for Data at Rest and in Transit

high

As the system, I want to encrypt all sensitive data both at rest and in transit, so that user data is protected from unauthorized access.

Dependencies:

9c9f6ceb... defa1f08...

Acceptance Criteria:

- ✓ Technical: All data stored in databases and S3 is encrypted with AES-256.
- ✓ Technical: All API and service communications use TLS 1.3 encryption.
- ✓ Functional: Encryption keys are rotated regularly and managed securely.
- ✓ Technical: Encryption status is audited and logged.
- ✓ Functional: No unencrypted sensitive data is stored or transmitted.

Story Points: 5

Estimated Effort: 8 hours

High Performance and Low Latency

This epic focuses on optimizing system performance to provide fast response times for real-time alerts, AI summaries, and financial data updates, ensuring a smooth user experience.

3 stories

Ensure Quick Data Synchronization Across Services

high

As the system, I want to ensure quick and consistent data synchronization across microservices, so that users see up-to-date financial information.

Dependencies:

29cade7d... e62a3e16... a2755730...

Acceptance Criteria:

- ✓ Technical: Data synchronization latency between services is under 1 second.
- ✓ Technical: Event-driven architecture with Kafka ensures reliable message delivery.
- ✓ Functional: Data consistency is maintained across services and UI.
- ✓ Technical: Synchronization failures are logged and retried automatically.
- ✓ Functional: Users see real-time updates on transactions and alerts.

Story Points: 5

Estimated Effort: 8 hours

Optimize AI Model Inference Latency

medium

As the system, I want to optimize AI model inference latency, so that natural language summaries and recommendations are generated quickly.

Dependencies:

8811015c... 59c39d6b...

Acceptance Criteria:

- ✓ Technical: AI model inference latency is under 2 seconds for 95% of requests.
- ✓ Technical: Caching and batching strategies are implemented for API calls.
- ✓ Functional: Users receive timely AI-generated insights without delay.
- ✓ Technical: Model serving infrastructure is scaled appropriately.
- ✓ Functional: Latency metrics are monitored and alerts configured.

Story Points: 3

Estimated Effort: 5 hours

Optimize API Response Times and Backend Algorithms

high

As the system, I want to optimize API response times and backend algorithms, so that users experience fast and responsive interactions.

Dependencies:

29cade7d... ab3bd87e... 15780369...

Acceptance Criteria:

- ✓ Technical: API response times are under 200ms for 90% of requests.
- ✓ Technical: Backend algorithms for categorization and scoring are optimized for performance.

- ✓ Functional: Users experience no noticeable lag during dashboard interactions.
- ✓ Technical: Profiling and code optimization are performed regularly.
- ✓ Functional: Performance improvements are validated with load testing.

Story Points: 5

Estimated Effort: 8 hours

Compliance with PCI DSS and GDPR Standards

This epic ensures the platform complies with PCI DSS and GDPR standards to protect user data and privacy, including encryption, consent management, and security audits.

3 stories

Conduct Regular Security Audits and Vulnerability Assessments

high

As the compliance officer, I want regular security audits and vulnerability assessments to ensure ongoing compliance with PCI DSS and GDPR standards.

Dependencies:

9c9f6ceb... defa1f08...

Acceptance Criteria:

- ✓ Functional: Quarterly security audits are scheduled and executed.
- ✓ Functional: Vulnerability scans and penetration tests are performed regularly.
- ✓ Technical: Findings are documented and tracked to resolution.
- ✓ Functional: Compliance reports are generated for regulatory review.
- ✓ Technical: Audit logs are maintained securely and accessible for review.

Story Points: 3

Estimated Effort: 5 hours

Implement User Data Control and Consent Management

high

As a registered user, I want to control my data and provide explicit consent for data collection, so that my privacy rights are respected.

Dependencies:

fb5d5c6c... defa1f08...

Acceptance Criteria:

- ✓ Functional: Users can view, edit, and revoke consent for data processing.
- ✓ Functional: Users can request data export and deletion (right to be forgotten).
- ✓ Technical: Consent status is stored and enforced in data processing workflows.
- ✓ Functional: Privacy policy and consent forms are presented clearly during onboarding.
- ✓ Technical: Compliance with GDPR and CCPA regulations is documented.

Story Points: 5

Estimated Effort: 8 hours

Implement Data Encryption at Rest and In Transit

high

As the system, I want to encrypt all sensitive data at rest and in transit, so that user data is protected against unauthorized access.

Dependencies:

9c9f6ceb... 9e926815...

Acceptance Criteria:

- ✓ Technical: All data stored in PostgreSQL and S3 is encrypted using AES-256.
- ✓ Technical: All network communications use TLS 1.3 encryption.
- ✓ Functional: Encryption keys are managed securely with AWS KMS or Vault.
- ✓ Technical: Encryption status is monitored and logged.
- ✓ Functional: No sensitive authentication data is stored post-authorization.

Story Points: 5

Estimated Effort: 8 hours

Secure Authentication and Authorization System

This epic covers implementing secure user authentication and authorization mechanisms to protect sensitive financial data, including OAuth, multi-factor authentication, and role-based access control.

3 stories

Implement Role-Based Access Control (RBAC)

high

As the system, I want to enforce role-based access control, so that users and services have appropriate permissions to protect sensitive data.

Dependencies:

defa1f08...

Acceptance Criteria:

- ✓ Functional: Roles and permissions are defined for users and services.
- ✓ Technical: Access control enforced at API Gateway and service layers.
- ✓ Functional: Unauthorized access attempts are logged and blocked.
- ✓ Technical: RBAC policies are configurable and auditable.
- ✓ Functional: Admin users have elevated access with audit trails.

Story Points: 5

Estimated Effort: 8 hours

Implement Multi-Factor Authentication (MFA)

medium

As a security-conscious user, I want to enable multi-factor authentication, so that my account is more secure.

Dependencies:

defa1f08...

Acceptance Criteria:

- ✓ Functional: Users can enable MFA via authenticator apps or SMS.
- ✓ Functional: MFA is required on login after enabling.
- ✓ Technical: QR codes are generated for authenticator app setup.
- ✓ Functional: Backup codes are provided for account recovery.
- ✓ Technical: MFA setup and verification are logged for audit.

Story Points: 5

Estimated Effort: 8 hours

Implement OAuth 2.0 and JWT-Based Authentication

high

As a registered user, I want to securely authenticate using OAuth 2.0 and JWT tokens, so that my account and data are protected.

Acceptance Criteria:

- ✓ Functional: Users can register and login using secure OAuth 2.0 flows.
- ✓ Technical: JWT tokens are issued with appropriate claims and expiration.
- ✓ Functional: Sessions are managed securely with token revocation support.
- ✓ Technical: Authentication service integrates with third-party identity providers.
- ✓ Functional: Login attempts are rate-limited and logged.

Story Points: 8

Estimated Effort: 13 hours

Scalable and Reliable Backend Infrastructure

This epic covers building a backend system capable of handling secure financial data processing, real-time alerts, and AI computations at scale with failover and redundancy.

2 stories

Implement Robust Database and Caching Layers

high

As the system, I want to implement a robust database and caching layer to support high throughput and low latency for financial data processing.

Dependencies:

9c9f6ceb...

Acceptance Criteria:

- Technical: PostgreSQL database deployed with multi-AZ replication and read replicas.
- Technical: Redis caching layer implemented for session and hot data caching.
- Functional: Database supports partitioning and indexing for large datasets.
- Technical: Backup and restore procedures are automated and tested.
- Functional: Database and cache performance meet latency requirements.

Story Points: 5

Estimated Effort: 8 hours

Design and Implement Cloud-Based Scalable Architecture

high

As the system, I want to deploy backend services on a cloud platform with auto-scaling and load balancing, so that the app can handle high concurrent user volumes reliably.

Acceptance Criteria:

- Technical: Backend services deployed on AWS ECS/EKS with auto-scaling enabled.
- Technical: Load balancers distribute traffic evenly with health checks.
- Functional: System supports 10,000 concurrent users with <1s response time.
- Technical: Services are containerized using Docker with multi-stage builds.
- Functional: Failover mechanisms ensure 99.99% uptime.

Story Points: 8

Estimated Effort: 13 hours

Bill Negotiation Service Integration

This epic covers integrating with bill negotiation services like BillShark API to help users lower recurring charges by negotiating bills such as internet or phone bills.

1 story

Integrate BillShark API for Bill Negotiation

medium

As the system, I want to integrate with BillShark API to initiate and track bill negotiation requests, so that users can reduce recurring expenses automatically.

Dependencies:

a2755730... 1dbe4dab...

Acceptance Criteria:

- Technical: API authentication and request handling implemented securely.
- Functional: Users can initiate bill negotiation requests from the app.
- Technical: Negotiation status and savings are tracked and updated.
- Functional: Users receive notifications on negotiation progress and results.
- Technical: Handles API errors and retries with logging.

Story Points: 5

Estimated Effort: 8 hours

Credit Risk Analysis Integration

This epic covers integrating with credit score APIs to analyze user financial risk and incorporate it into financial scoring and recommendations.

1 story

Integrate Credit Score API for Risk Analysis

medium

As the system, I want to securely access credit score and risk data via Credit Score API, so that financial scoring and recommendations include risk factors.

Dependencies:

3935a7e6...

Acceptance Criteria:

- ✓ Technical: Secure API integration with OAuth 2.0 or API keys.
- ✓ Functional: Credit score and risk factors are retrieved and stored.
- ✓ Technical: Data privacy and compliance with GDPR ensured.
- ✓ Functional: Risk data influences financial score calculations.
- ✓ Technical: Handles API errors and rate limits gracefully.

Story Points: 5

Estimated Effort: 8 hours

Real-time Alert Messaging Integration

This epic covers integrating with messaging platforms like Twilio and WhatsApp API to send real-time alerts and notifications via SMS or chat apps.

1 story

Integrate Twilio and WhatsApp APIs for Alert Messaging

high

As the system, I want to send alerts and notifications via Twilio SMS and WhatsApp APIs, so that users receive timely financial alerts on their preferred channels.

Dependencies:

a2755730...

Acceptance Criteria:

- ✓ Technical: API integration with secure authentication and key management.
- ✓ Functional: Supports multiple messaging channels with user preference respect.

- ✓ Technical: Implements message queue with retry and dead-letter handling.
- ✓ Functional: Delivery status and errors are tracked and logged.
- ✓ Technical: Supports webhook callbacks for message status updates.

Story Points: 5

Estimated Effort: 8 hours

Natural Language Processing Integration

This epic covers integrating with natural language AI services like OpenAI to generate plain-English financial summaries and recommendations.

1 story

Integrate OpenAI API for Natural Language Generation

medium

As the system, I want to integrate with OpenAI API to generate natural language financial summaries and recommendations, so that users receive clear and personalized insights.

Acceptance Criteria:

- ✓ Technical: API integration with secure key management.
- ✓ Functional: Supports customizable summary generation based on user data.
- ✓ Technical: Latency optimized for real-time responses (<2 seconds).
- ✓ Functional: Handles API failures with fallback messaging.
- ✓ Technical: Logs API usage and errors for monitoring.

Story Points: 5

Estimated Effort: 8 hours

Investment Tracking Integration

This epic covers integrating with investment data APIs like Yahoo Finance to track user investment accounts and provide insights.

1 story

Integrate Yahoo Finance API for Investment Account Data

medium

As the system, I want to securely access investment account data via Yahoo Finance API, so that users can monitor their investments within the app.

Acceptance Criteria:

- ✓ Technical: Secure API authentication implemented.
- ✓ Functional: Supports multiple investment platforms and account types.
- ✓ Technical: Data aggregation and normalization is performed.
- ✓ Functional: Investment data is updated regularly and available to users.

- Technical: Handles API errors and rate limits gracefully.

Story Points: 5

Estimated Effort: 8 hours

Subscription Billing Detection Integration

This epic covers integrating with subscription billing detection services like Stripe Billing to identify active subscriptions from user financial data accurately.

1 story

Integrate Stripe Billing API for Subscription Detection

high

As the system, I want to integrate with Stripe Billing API to detect active subscriptions and retrieve subscription metadata, so that subscription management is accurate.

Acceptance Criteria:

- Technical: API authentication and secure data retrieval implemented.
- Functional: Subscription metadata including status, amount, and frequency is retrieved.
- Technical: Data normalization ensures consistency with internal subscription models.
- Functional: Integration supports webhook events for subscription updates.
- Technical: Error handling and retry mechanisms are in place.

Story Points: 5

Estimated Effort: 8 hours

Banking Data Access Integration

This epic covers integrating with secure banking data access services like Plaid to connect user bank accounts, credit cards, and investment accounts securely and reliably.

1 story

Integrate Plaid API for Secure Banking Data Access

high

As the system, I want to integrate with Plaid API using OAuth 2.0, so that user financial data can be securely accessed and synchronized.

Acceptance Criteria:

- Technical: OAuth 2.0 authentication implemented for Plaid API.
- Functional: System retrieves account and transaction data from multiple financial institutions.
- Technical: Handles API rate limits, errors, and retries gracefully.
- Functional: Data synchronization status is tracked and logged.
- Technical: Integration supports webhook callbacks for real-time updates.

Story Points: 8

Estimated Effort: 13 hours

Automated Decisioning with User Approval

This epic supports automated financial actions like subscription cancellations and bill negotiations but requires explicit user approval before execution, with status tracking and notifications.

3 stories

Track and Notify Users of Automated Action Status

medium

As a registered user, I want to receive updates on the status of my approved automated actions, so that I am informed of progress and outcomes.

Dependencies:

1dbe4dab...

Acceptance Criteria:

- ✓ Functional: Users receive real-time status updates (pending, in progress, completed, failed).
- ✓ Functional: Status updates are accessible in the app's action history section.
- ✓ Technical: System polls external APIs for action status and updates database accordingly.
- ✓ Functional: Users receive notifications for completed or failed actions.
- ✓ Technical: Audit logs capture all action events for compliance.

Story Points: 3

Estimated Effort: 5 hours

Develop User Approval Workflow for Automated Actions

medium

As a registered user, I want to approve or reject automated financial actions before they are executed, so that I maintain control over my finances.

Dependencies:

d16960bf...

Acceptance Criteria:

- ✓ Functional: Users receive notifications of recommended actions requiring approval.
- ✓ Functional: Users can approve or reject actions via the app UI.
- ✓ Technical: Approval status and timestamps are recorded in automated_actions table.
- ✓ Functional: System only executes actions after explicit user approval.
- ✓ Technical: Workflow supports audit logging and rollback.

Story Points: 5

Estimated Effort: 8 hours

Implement Automated Action Recommendation Engine

medium

As the system, I want to recommend automated financial actions based on AI insights, so that users can optimize their finances proactively.

Dependencies:

d400d5cb... a75379b9... 15780369...

Acceptance Criteria:

- ✓ Functional: System generates actionable recommendations for subscription cancellations and bill negotiations.
- ✓ Technical: Recommendations are stored and linked to user accounts.
- ✓ Functional: Recommendations include explanations and expected benefits.
- ✓ Technical: Integrates with AI Recommendation Engine and Subscription Management.
- ✓ Functional: Users can review recommendations before approval.

Story Points: 5

Estimated Effort: 8 hours

Demo Mode with Limited Functionality

This epic covers allowing users to explore the app with limited features without account creation to showcase value before signup, including guided tours and feature gating.

2 stories

Create Guided Tour for Demo Mode Users

low

As a demo mode user, I want a guided tour highlighting key features, so that I understand the app's value and how to use it.

Dependencies:

07c7b460...

Acceptance Criteria:

- ✓ Functional: Guided tour walks users through core features with sample data.
- ✓ Functional: Tour can be skipped or restarted.
- ✓ Technical: Tour content is configurable and localized.
- ✓ Functional: Tour encourages users to sign up for full access.
- ✓ Technical: Frontend React components with smooth UX.

Story Points: 2

Estimated Effort: 3 hours

Implement Demo Mode Environment and Session Management

low

As a prospective user, I want to try the app in demo mode without creating an account, so that I can evaluate its value before signing up.

Acceptance Criteria:

- ✓ Functional: Demo mode provides access to limited features with sample data.
- ✓ Technical: Demo sessions are tracked and expire after a set period.
- ✓ Functional: Users can transition from demo to full account creation seamlessly.
- ✓ Technical: Demo data is isolated and does not affect production data.
- ✓ Functional: Demo mode UI clearly indicates limited functionality.

Story Points: 3

Estimated Effort: 5 hours

User Feedback and AI Recommendation Correction

This epic includes allowing users to provide feedback on AI-generated recommendations and correct inaccuracies to improve system learning and AI accuracy over time.

2 stories

Integrate Feedback into AI Model Retraining Pipeline

medium

As the system, I want to incorporate user feedback into AI model retraining, so that recommendations improve over time.

Dependencies:

6b80107e...

Acceptance Criteria:

- ✓ Technical: Feedback data is processed and used to update training datasets.
- ✓ Technical: Retraining pipeline runs periodically with new feedback.
- ✓ Functional: AI recommendation accuracy improves based on feedback.
- ✓ Technical: Retraining process is logged and monitored.
- ✓ Functional: System supports rollback in case of degraded model performance.

Story Points: 5

Estimated Effort: 8 hours

Implement In-App Feedback Forms and Rating System

medium

As a registered user, I want to provide feedback and rate AI-generated recommendations, so that I can help improve the system's accuracy.

Dependencies:

8811015c... f28145cb...

Acceptance Criteria:

- Functional: Users can submit feedback and ratings on individual AI recommendations.
- Functional: Feedback forms are accessible and user-friendly.
- Technical: Feedback data is stored in user_feedback table.
- Functional: Users receive confirmation upon feedback submission.
- Technical: Frontend React components with validation and error handling.

Story Points: 3

Estimated Effort: 5 hours

Seamless Onboarding with Secure Account Connections

This epic covers guiding users through a secure and intuitive onboarding process to connect bank, credit card, and investment accounts, emphasizing data security and benefits.

2 stories

Implement Secure Authentication and Account Linking

high

As a new user, I want to securely authenticate and link my bank, credit card, and investment accounts, so that my financial data is safely accessed.

Acceptance Criteria:

- Functional: Users authenticate via OAuth 2.0 with Plaid and other providers.
- Technical: Authentication uses secure token storage and JWT for sessions.
- Functional: Account linking status and errors are clearly displayed.
- Technical: Backend supports token refresh and error handling.
- Functional: Supports multi-factor authentication during onboarding.

Story Points: 8

Estimated Effort: 13 hours

Design and Implement Step-by-Step Onboarding Tutorial

high

As a new user, I want a guided onboarding tutorial explaining how to connect my accounts and the benefits, so that I can start using the app confidently.

Acceptance Criteria:

- Functional: Tutorial guides users through account linking steps with clear instructions.
- Functional: Tutorial highlights data security and privacy practices.
- Technical: Tutorial content is configurable and localized.
- Functional: Users can skip or revisit tutorial anytime.
- Technical: Frontend React components with smooth UX.

Story Points: 3

Estimated Effort: 5 hours

Downloadable PDF Financial Reports

This epic covers generating and allowing users to download detailed PDF reports summarizing their financial status, including summaries, charts, and recommendations.

2 stories

Implement PDF Report Generation and Download

medium

As a registered user, I want to generate and download branded PDF financial reports, so that I can keep records and share my financial insights.

Dependencies:

0cc810d6...

Acceptance Criteria:

- ✓ Functional: Users can request report generation for selected periods.
- ✓ Functional: Reports include summaries, charts, and recommendations.
- ✓ Technical: PDF generation uses PDFKit and data visualization libraries.
- ✓ Functional: Reports are stored in S3 and downloadable via secure links.
- ✓ Technical: Report generation is asynchronous with status tracking.

Story Points: 5

Estimated Effort: 8 hours

Develop Financial Report Data Compilation Service

medium

As the system, I want to compile comprehensive financial data including transactions, budgets, subscriptions, and AI summaries, so that reports can be generated accurately.

Dependencies:

673302d5... 8811015c...

Acceptance Criteria:

- ✓ Functional: Data compilation supports multiple report periods (monthly, quarterly, custom).
- ✓ Technical: Data is aggregated efficiently with minimal latency.
- ✓ Technical: Supports inclusion of charts and AI-generated summaries.
- ✓ Functional: Data is validated for completeness before report generation.
- ✓ Technical: Uses ReportDataCompiler component.

Story Points: 3

Estimated Effort: 5 hours

Investment and Savings Strategy Recommendations

This epic covers recommending personalized investment and savings strategies based on user financial data and goals, updating recommendations over time based on feedback and data changes.

3 stories

Implement User Profile and Goal Management for Recommendations

medium

As a registered user, I want to set and update my financial goals and preferences, so that recommendations are personalized and relevant.

Dependencies:

29cade7d...

Acceptance Criteria:

- ✓ Functional: Users can create, edit, and delete financial goals and preferences.
- ✓ Functional: Goals influence AI recommendation outputs.
- ✓ Technical: Goals stored securely and linked to user profiles.
- ✓ Functional: UI supports goal management with validation and feedback.
- ✓ Technical: Backend APIs support CRUD operations for goals.

Story Points: 5

Estimated Effort: 8 hours

Develop AI Recommendation Engine for Investment and Savings Strategies

medium

As a registered user, I want personalized investment and savings recommendations based on my financial data and goals, so that I can optimize my financial growth.

Dependencies:

f28145cb... 15780369... 8811015c...

Acceptance Criteria:

- ✓ Functional: AI engine generates tailored recommendations considering user goals and risk profile.
- ✓ Functional: Recommendations update dynamically based on user feedback and data changes.
- ✓ Technical: Uses AI Recommendation Engine integrated with OpenAI API.
- ✓ Functional: Users can provide feedback to improve recommendation accuracy.
- ✓ Technical: Recommendations stored and versioned in ai_recommendations table.

Story Points: 8

Estimated Effort: 13 hours

Integrate Investment Data Aggregation

medium

As the system, I want to aggregate user investment account data from multiple platforms, so that I can provide comprehensive investment insights.

Dependencies:

29cade7d...

Acceptance Criteria:

- ✓ Functional: Investment data is retrieved securely from Yahoo Finance API and other providers.
- ✓ Technical: Data is normalized and stored in investments table.
- ✓ Functional: Users can view current holdings and performance in the app.
- ✓ Technical: Data sync supports incremental updates and error handling.
- ✓ Functional: Supports multiple investment account types.

Story Points: 5

Estimated Effort: 8 hours

Financial Score Generation Based on Habits

This epic covers automatically generating a financial score reflecting users' financial habits and health, displaying the score with explanations and improvement tips.

2 stories

Display Financial Score and Improvement Tips in UI

medium

As a registered user, I want to see my financial score with explanations and tips, so that I understand my financial health and how to improve it.

Dependencies:

3935a7e6...

Acceptance Criteria:

- ✓ Functional: Score and tips are displayed prominently on the dashboard.
- ✓ Functional: Users can view detailed breakdown of score factors.
- ✓ Technical: Frontend uses React components with dynamic data fetching.
- ✓ Functional: Score updates in real-time or on refresh.
- ✓ Technical: UI supports accessibility standards (WCAG 2.1 AA).

Story Points: 3

Estimated Effort: 5 hours

Define and Implement Financial Scoring Algorithm

medium

As the system, I want to calculate a financial score based on spending and saving habits, so that users have an easy metric to gauge their financial wellness.

Dependencies:

29cade7d... ab3bd87e... f4446b88...

Acceptance Criteria:

- ✓ Functional: Score reflects multiple factors including spending, saving, and subscription management.
- ✓ Technical: Algorithm runs daily and updates scores in financial_scores table.
- ✓ Functional: Score is normalized to a 0-100 scale with clear interpretation.
- ✓ Technical: Factors and tips are stored in JSONB format for flexibility.
- ✓ Functional: Users can view score history and improvement tips.

Story Points: 5

Estimated Effort: 8 hours

Predictive Monthly Spend and Low Balance Warnings

This epic covers predicting users' monthly spending based on historical data and warning them proactively about potential low balances to help avoid overdrafts and improve financial planning.

2 stories

Trigger Low Balance Warnings Based on Predictions

medium

As the system, I want to trigger alerts warning users about potential low balances before critical thresholds, so that users can avoid overdrafts.

Dependencies:

ebb86351... 15780369... a2755730...

Acceptance Criteria:

- ✓ Functional: Alerts are generated when predicted spend exceeds available balance thresholds.
- ✓ Functional: Users receive warnings via preferred notification channels.
- ✓ Technical: LowBalanceWarningEngine integrates with AlertDispatcher.
- ✓ Functional: Users can configure threshold sensitivity in settings.
- ✓ Technical: Alerts include actionable recommendations.

Story Points: 5

Estimated Effort: 8 hours

Develop Predictive Analytics Model for Monthly Spending

medium

As the system, I want to analyze historical spending data to predict upcoming monthly expenses, so that users can plan their finances better.

Dependencies:

29cade7d... ab3bd87e...

Acceptance Criteria:

- ✓ Functional: Model predicts monthly spending with confidence scores.
- ✓ Technical: Model retrains periodically using latest transaction data.
- ✓ Functional: Predictions are stored and accessible via API.
- ✓ Technical: Uses Python ML frameworks integrated with backend.
- ✓ Functional: Predictions support multiple user segments and categories.

Story Points: 8

Estimated Effort: 13 hours

Plain-English Financial Health Summaries

This epic covers generating easy-to-understand natural language summaries of users' financial health, spending patterns, and AI-driven recommendations, presented in-app and in downloadable reports.

3 stories

Present Financial Summaries in App and Reports

medium

As a registered user, I want to view AI-generated financial summaries in the app and in downloadable PDF reports, so that I can understand and share my financial health easily.

Dependencies:

8811015c… 3b92daa3…

Acceptance Criteria:

- ✓ Functional: Summaries are displayed in a dedicated dashboard section with charts and text.
- ✓ Functional: PDF reports include summaries with charts and recommendations.
- ✓ Technical: Frontend uses React and Material-UI for presentation.
- ✓ Functional: Users can download reports from the app.
- ✓ Technical: Reports are generated asynchronously and stored in S3.

Story Points: 5

Estimated Effort: 8 hours

Integrate OpenAI for Natural Language Summary Generation

medium

As the system, I want to generate plain-English financial summaries and recommendations using AI, so that users can easily understand their financial status.

Dependencies:

673302d5…

Acceptance Criteria:

- ✓ Functional: Summaries include spending patterns, budget status, and actionable recommendations.
- ✓ Technical: OpenAI API is integrated with latency under 2 seconds for summary generation.
- ✓ Functional: Summaries are available in-app and included in downloadable reports.
- ✓ Technical: Supports customization of summary tone and detail level.
- ✓ Functional: Users can request updated summaries on demand.

Story Points: 5

Estimated Effort: 8 hours

Aggregate Financial Data for Summary Generation

medium

As the system, I want to aggregate user financial data and spending patterns, so that I can prepare inputs for natural language summary generation.

Dependencies:

29cade7d... d400d5cb...

Acceptance Criteria:

- ✓ Functional: Aggregated data includes categorized expenses, budgets, subscriptions, and investments.
- ✓ Technical: Aggregation runs on schedule and on-demand with low latency.
- ✓ Technical: Data is stored temporarily for summary generation.
- ✓ Functional: Aggregation supports multiple time periods (daily, weekly, monthly).
- ✓ Technical: Uses FinancialSummaryAggregator component.

Story Points: 3

Estimated Effort: 5 hours

Real-time Alerts and Bill Reminders

This epic covers sending customizable alerts and reminders for upcoming bills, unusual spending, low balances, and subscription changes via multiple notification channels including in-app, SMS, and WhatsApp.

4 stories

Create Bill Reminder Alerts and Notifications

high

As a registered user, I want to receive timely reminders for upcoming bills and payments, so that I avoid late fees and missed payments.

Dependencies:

a75379b9... a2755730...

Acceptance Criteria:

- ✓ Functional: System generates alerts for bills due within configurable time windows.

- Functional: Users receive reminders via their preferred channels.
- Technical: Alerts link to bill details and payment options in the app.
- Functional: Users can snooze or reschedule reminders.
- Technical: Bill due dates and statuses are synced from bill negotiation and subscription services.

Story Points: 3

Estimated Effort: 5 hours

Implement Multi-Channel Alert Dispatching

high

As the system, I want to send alerts via in-app notifications, SMS, and WhatsApp according to user preferences, so that users receive notifications on their preferred channels.

Dependencies:

15780369... e62a3e16...

Acceptance Criteria:

- Functional: Alerts are sent via in-app, SMS (Twilio), and WhatsApp APIs.
- Functional: Alerts respect user preferences for type, frequency, and channel.
- Technical: AlertDispatcher service queues and sends messages with retry and dead-letter handling.
- Functional: Users can dismiss alerts in-app and mark them as read.
- Technical: Alert delivery status is tracked and logged.

Story Points: 5

Estimated Effort: 8 hours

Develop Real-Time Event Detection for Alerts

high

As the system, I want to detect events such as upcoming bills, unusual spending, and low balances in real-time, so that timely alerts can be sent to users.

Dependencies:

29cade7d... 95e874eb...

Acceptance Criteria:

- Functional: System detects bill due dates, unusual charges, and low balance thresholds in real-time.
- Technical: EventMonitor service processes transaction and subscription data streams.
- Technical: Thresholds and anomaly detection algorithms are configurable.
- Functional: Detected events trigger alert creation in alerts table.
- Technical: Event detection latency is under 1 second.

Story Points: 5

Estimated Effort: 8 hours

Implement User Alert Preferences Management

high

As a registered user, I want to customize the types and frequency of alerts I receive, so that I only get relevant notifications.

Dependencies:

29cade7d...

Acceptance Criteria:

- ✓ Functional: Users can enable or disable alert types and set frequency preferences (real-time, daily, weekly).
- ✓ Functional: Users can select preferred notification channels (in-app, SMS, WhatsApp).
- ✓ Technical: Preferences are stored securely in user_alert_preferences table.
- ✓ Functional: Changes to preferences are saved and reflected immediately.
- ✓ Technical: Frontend UI built with React and Redux.

Story Points: 3

Estimated Effort: 5 hours

Subscription Detection and Management

This epic includes detecting active subscriptions from user financial data, identifying duplicates or unusual charges, providing AI recommendations for cancellations, and managing user-approved subscription cancellations with status tracking.

5 stories

Display Subscription Management UI with Duplicate and Unusual Charge Indicators

high

As a registered user, I want to view my subscriptions with clear indicators for duplicates and unusual charges, so that I can easily identify subscriptions to review.

Dependencies:

a75379b9... 95e874eb...

Acceptance Criteria:

- ✓ Functional: Subscription list shows all active subscriptions with duplicate and unusual flags.
- ✓ Functional: Users can filter and sort subscriptions by status and flags.
- ✓ Technical: UI built using React and Material-UI components.
- ✓ Functional: Users can initiate cancellation or negotiation actions from the UI.
- ✓ Technical: Frontend consumes subscription management APIs with proper error handling.

Story Points: 5

Estimated Effort: 8 hours

Implement Subscription Cancellation Workflow with User Approval

high

As a registered user, I want to approve subscription cancellations before they are executed, so that I maintain control over my financial decisions.

Dependencies:

a75379b9... d400d5cb...

Acceptance Criteria:

- ✓ Functional: Users can initiate cancellation requests from the subscription management UI.
- ✓ Functional: System requires explicit user confirmation before executing cancellations.
- ✓ Technical: Cancellation requests and statuses are tracked in subscription_cancellation_requests table.
- ✓ Functional: Users receive status updates on cancellation progress.
- ✓ Technical: Workflow uses Saga pattern to manage multi-step cancellation process.

Story Points: 8

Estimated Effort: 13 hours

Provide AI Recommendations for Subscription Cancellations

high

As a registered user, I want to receive AI-driven recommendations for subscription cancellations, so that I can reduce unnecessary expenses.

Dependencies:

a75379b9...

Acceptance Criteria:

- ✓ Functional: AI engine generates cancellation recommendations based on subscription usage and spending patterns.
- ✓ Functional: Recommendations are presented clearly in the subscription management UI.
- ✓ Technical: Recommendations are stored and updated in the ai_recommendations table.
- ✓ Functional: Users can provide feedback on recommendations to improve AI accuracy.
- ✓ Technical: Integrates with OpenAI API for natural language explanations.

Story Points: 5

Estimated Effort: 8 hours

Identify Duplicate and Unusual Subscription Charges

high

As the system, I want to identify duplicate or unusual subscription charges, so that users can be alerted to potential wasteful spending.

Dependencies:

a75379b9...

Acceptance Criteria:

- ✓ Functional: System flags duplicate subscriptions based on merchant and amount similarity.

- Functional: System detects unusual subscription charges based on historical spending patterns.
- Technical: Duplicate and unusual flags are stored in subscription records.
- Functional: Users receive alerts about duplicates and unusual charges.
- Technical: Detection algorithms run as batch jobs with configurable thresholds.

Story Points: 5

Estimated Effort: 8 hours

Detect Active Subscriptions from Financial Data

high

As the system, I want to automatically detect active subscriptions from user transaction data, so that users can manage their recurring expenses effectively.

Dependencies:

29cade7d...

Acceptance Criteria:

- Functional: System identifies recurring transactions as subscriptions with at least 90% accuracy.
- Technical: Subscription detection algorithm runs daily and updates subscription records.
- Technical: Detected subscriptions are stored with metadata including merchant, amount, and frequency.
- Functional: Users can view detected subscriptions in the app.
- Technical: Uses Stripe Billing API integration for enhanced subscription metadata.

Story Points: 8

Estimated Effort: 13 hours

Automatic Expense Categorization and Budget Tracking

This epic covers the automatic categorization of user expenses from connected financial accounts and tracking of budgets to monitor spending habits and financial health. It includes data ingestion, ML-based categorization, budget management, and real-time updates.

4 stories

Provide Real-Time Updates on Spending and Budget Status

high

As a registered user, I want to see real-time updates on my spending and budget status in the app, so that I can make timely financial decisions.

Dependencies:

29cade7d... ab3bd87e... f4446b88...

Acceptance Criteria:

- Functional: Dashboard displays up-to-date spending and budget usage data.

- ✓ Technical: Real-time data updates are pushed via WebSocket or polling mechanisms.
- ✓ Technical: Backend supports low-latency queries (<200ms) for dashboard data.
- ✓ Functional: Users can filter spending by category and date range.
- ✓ Technical: Frontend uses React with Redux Toolkit for state management.

Story Points: 5

Estimated Effort: 8 hours

Develop Budget Tracking and Limit Monitoring

high

As a registered user, I want to set budgets for expense categories and track spending against these budgets, so that I can monitor and control my financial health.

Dependencies:

29cade7d... ab3bd87e...

Acceptance Criteria:

- ✓ Functional: Users can create, edit, and delete budgets for categories with specified limits and periods.
- ✓ Functional: System tracks spending in real-time and compares it against budgets.
- ✓ Functional: Users receive notifications when spending approaches or exceeds budgets.
- ✓ Technical: Budgets and tracking data are stored in PostgreSQL with proper indexing.
- ✓ Technical: Budget tracking integrates with transaction categorization updates.

Story Points: 5

Estimated Effort: 8 hours

Implement Machine Learning Model for Expense Categorization

high

As the system, I want to automatically categorize user transactions using ML models, so that users can understand their spending habits without manual input.

Dependencies:

29cade7d...

Acceptance Criteria:

- ✓ Functional: Transactions are categorized into predefined categories with at least 85% accuracy.
- ✓ Technical: ML model is integrated into the Expense Categorization Service.
- ✓ Technical: Categorization results are stored and updated in the transactions table.
- ✓ Functional: Users can manually correct categories which update the model training data.
- ✓ Technical: Model retraining pipeline supports incremental learning from user feedback.

Story Points: 8

Estimated Effort: 13 hours

Setup Financial Account Data Ingestion

high

As the system, I want to ingest transaction data from connected bank, credit card, and investment accounts, so that user expenses can be tracked and categorized automatically.

Acceptance Criteria:

- Functional: System ingests transaction data from connected accounts daily or on-demand.
- Technical: Uses Plaid API with OAuth 2.0 authentication for secure data retrieval.
- Technical: Data ingestion handles errors with retries and logs failures.
- Technical: Transactions are normalized and stored in PostgreSQL with appropriate indexing.
- Functional: Users can view imported transactions in their account.

Story Points: 5

Estimated Effort: 8 hours

Spike Stories

Spike: Performance Testing and Optimization Research

Spike

high

As a performance engineer, I need to research best practices and tools for load testing and optimizing the AI FinanceMaster app, so that we can meet performance SLAs.

Timebox: 4 days

Expected Outcomes:

- Performance testing plan
- Tool selection and configuration
- Initial load test results and bottleneck identification

Acceptance Criteria:

- Testing plan approved
- Tools integrated into CI/CD
- Baseline performance metrics collected

Story Points: 5

Spike: Research Integration Patterns for Bill Negotiation APIs

Spike

medium

As a backend engineer, I need to explore integration patterns and workflows for bill negotiation APIs like BillShark, so that we design a robust and user-friendly negotiation feature.

Timebox: 3 days

Expected Outcomes:

- Integration design document
- Sample API calls and response handling
- Workflow diagrams for negotiation process

Acceptance Criteria:

- Integration approach documented
- Sample API integration tested
- Workflow validated with product team

Story Points: 3

Spike: Evaluate AI Model Approaches for Expense Categorization Spike high

As a data scientist, I need to research and evaluate different machine learning models for expense categorization, so that we select the most accurate and scalable approach.

Timebox: 1 week

Expected Outcomes:

- Comparison report of ML models with accuracy metrics
- Proof-of-concept implementation of top models
- Recommendation for production model

Acceptance Criteria:

- Research completed with documented findings
- POC models tested on sample data
- Model selection justified based on accuracy and performance

Story Points: 5