[FED, BED, FSD] - Technical Specifications

Document Purpose

A technical specifications document outlines the details of a technical product or system, including its functions, capabilities, and design. They keep your team working on the same goals with a firm idea of how to go about it.

Header definition for the developer:

- [FED] = Front End Development
- [BED] = Back End Development
- [FSD] - End to End Development (Fullstack)

Background

Story/Kanban/Feature Abstract

Who: [The user or system] that will interact with this tech specification

What: The user or systems intent

Why: The overall benefit or goal

[Most of this is defined in the business requirements or use cases, this description just gives the developer some additional context].

Testing and Validation

Detail out some things you know both the Dev and QA should hit to adequately and independently confirm the intended behaviors of the ticket. Some items to consider:

- Queue or Environment variable names
- Logs item property names
- Table or columns in a DB
- Specific URLs
- APIs

Test Coverage

1. Device Type (Mobile, Browser, etc.)

2. OS Version
3. Is backwards-compatibility testing required?

Test Scenarios

	Scenario	Acceptance Criteria

1	Scenario description [Should be written as a stand alone executable event].	• `Given` [steps are used to describe the initial context of the system - the *scene* of the scenario.]
		  ◦ `And` some other precondition
		• `When` [steps are used to describe an event, or an *action*.]
		  ◦ `And` some other condition
		• `Then` [steps are used to describe an *expected* outcome, or result.]
		
		Reference site for writing **Gherkin/Cucumber/BDD** scripts.

2	Additional Scenario(s) description	

Implementation Checklist

Work Item # (Agile Story #, Kanban Ticket# or Waterfall Requirement)	[If you have a separate system or using Jira, paste a copy of the link here for a developer to reference]
Post-Release Dev	[At times there are tasks that need to be completed for a work item once the release to production has been completed. Use this section to detail out these tasks]
Stakeholders	Area of business advocating for the specification: • `Management` • `Accounting` • `Warehouse` • `Shipping`

Verification Criteria (Success Measurement)	How can we officially verify that this work item produced the effect you were intending when you wrote up the tech specs for it. If SQL is how you plan to verify it, put the query in here:
	`select *`
	`from some table`
	If there is some report or KPI that needs to be used to measure success, please detail out that report or link to the report requirements.

Executable(s) or App(s)	Mention the component(s) involved in the context of this spec • `api name` • `message name` • `site name` • `app name`
Trunk Based Development	This is for the developer to update. If applicable, detail this work item would allow or block the deployment branch that the item is in to production. This is a driver for us when we are deciding what can stay or needs to be pulled out of a branch when creating releases. `TBD-Allow` - denotes that dev work is related to this ticket can be included in a production release even if it hasn't passed testing and acceptance. If allow, please detail the reason. `TBD-Block` - denotes that dev work related to this ticket can't be included in a production release until it has passed testing and acceptance.

URL(s) / Queue(s) / APIs	Detail some indication of which URLs or Queues are involved in the work item. Specifically note if any should be created as part of this work item, other wise it's assumed that you intend for existing behavior to change.
	Example of how this section should be used:
	`Azure Storage/documentstoreage/Queues/doc-api-queue // existing queue that accepts Documentapi requests`
	`GET /api/document/{documentid} // Use to get a document for download`
	`POST /api/document/{documentid} // New, to create/change a document`
	`Azure Funcs/PCFunctions/AnyDups // job that checks for duplicate records to the team for manual intervention`

Authentication	Specify a human or system that interacts with this work item will prove who they are.
Authorization	Specify how the human or system that interacts with this work item should be evaluated to know whether they should be allows to do what they are wanting to do.
Integration	Specify, with brief details, if there is any integration involved with other executables or apps.
Fault Strategy	How do we want to handle failures? These are decisions that should be confirmed by the writer of the tech spec. Here are some examples: • **Queue up request** - often used with retry/dlq. Use this approach if you want the user to not have to wait for the operation's outcome. You will need to figure out some

	operational process to handle errors, since the user wouldn't know that the action didn't work unless we take some specific action. • **Retry** - Used every time the Queue up request option is used but can be used to block operation while the user waits for the software to work. • **DLQ/Notify** - DLQ means Dead Letter Queue, a historical term, but this is a strategy to notify some team when a failure occurs and that the operational team will need to take some manual actions. • **Fail Visibly** - show the end user the failure. Show them a page when we hit a 404. If you don't know what to document, contact the dev or write a note to dev to contact you when they run across failures during coding and unit testing.
Developer Checklist	
Maintenance & Support	Provide a description of the maintenance and support procedures that will be used to keep the product or system running smoothly and address any issues that may arise.

Specification Details

This section details the work we need to accomplish, including happy path, alternate path and exception paths. Detail out the need to connect to external systems, desired database stats (new tables/columns/relationships) to record for the business need. The "normal" tech specs should be laid out in whatever way the writer wants to write them. Your audience is the developer. In my past, developers want clear cut requirements in this section but if you need to write some dev considerations and disclaimers, please call them out using info panels. Your job as the writer is to write what's needed and let the developer figure out how. The normal layout for technical specifications follows this structure:

> This requirement is up for consideration from a dev as we don't fully understand the how.

Functionality Overview

Provide a detailed description of the product's or system's functions and capabilities, including any required inputs, outputs, and performance criteria.

Technical &/or Design Requirements

Provide a detailed description of technical requirements, including hardware, software, network, and security requirements. Also include the product's or system's design, including any diagrams, schematics, or illustrations that help explain its architecture and components.

	Requirement Description	Comments
1		
2		

Some small examples of tech specs

1. Create a new service to check the database for an existing record.
    a. Service Name: Existing Record

b.
```
GET /api/existing record?[some parameter]
```

2. Using the service call `database.table` and bounce the parameter value in the service against `table.somefield`.
3. If a record is found return a 200 http response message.

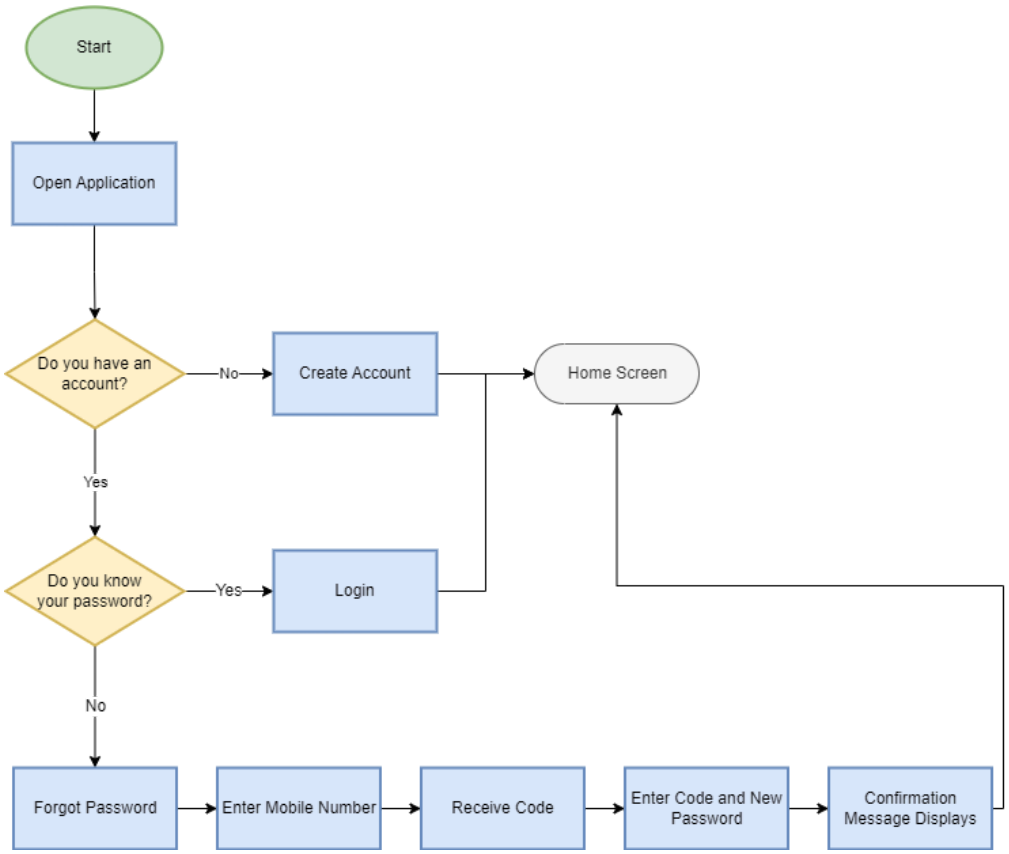
a.
```
// GET  /api/existingrecord
// parameter
{
  "[some parameter]"
}

// 200 response
  "record id": "[some value]"                          // the id of the existing record

//204 response
No response body needed
```

4. If the record is not found return a 204 http response with no response body.

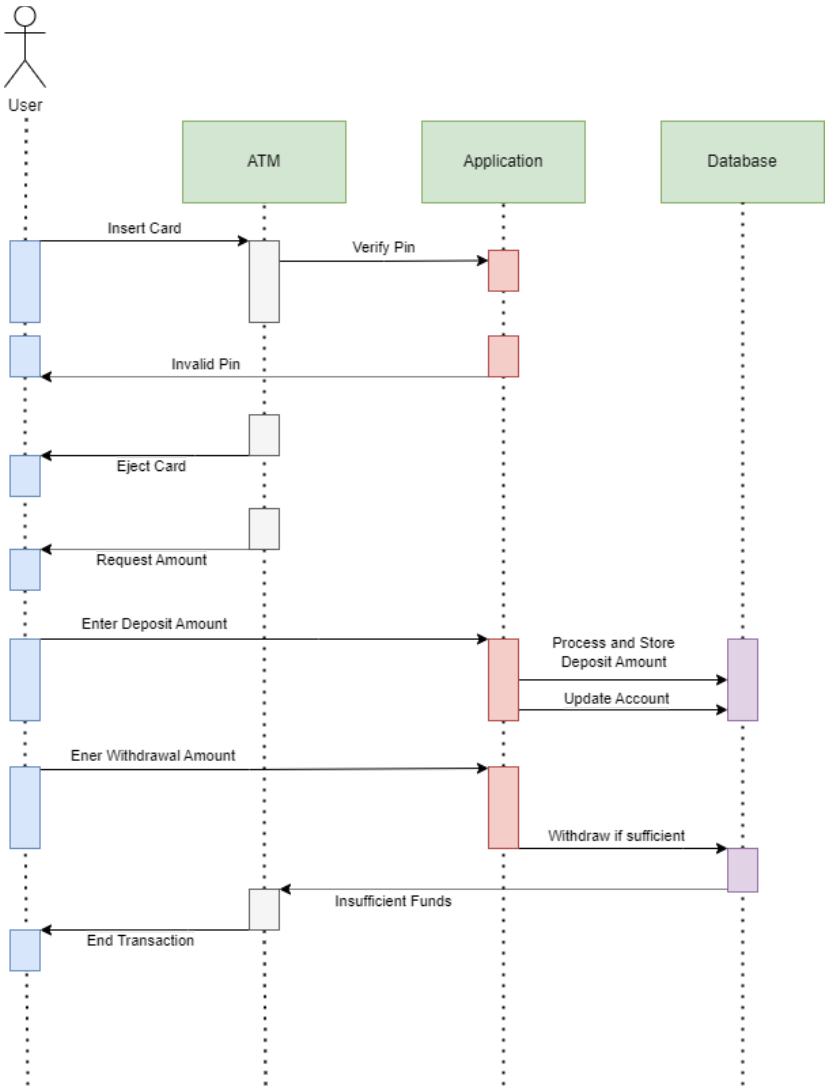
At other times a table needs to be created. Reference the DB Table template and paste the table format here.

Supporting Documents

System Flow Diagram Example



Sequence Diagram Example


User
ATM
Application
Database
Insert Card
Verify Pin
Invalid Pin
Eject Card
Request Amount
Enter Deposit Amount
Process and Store
Deposit Amount
Update Account
Ener Withdrawal Amount
Withdraw if sufficient
Insufficient Funds
End Transaction