Backend Developer Test – ATM Simulation

Prepared By OMID VAHDANI

Instruction to run the application:
Requirements:

- Apache Server.
- MySql Database.
- Web Browser.

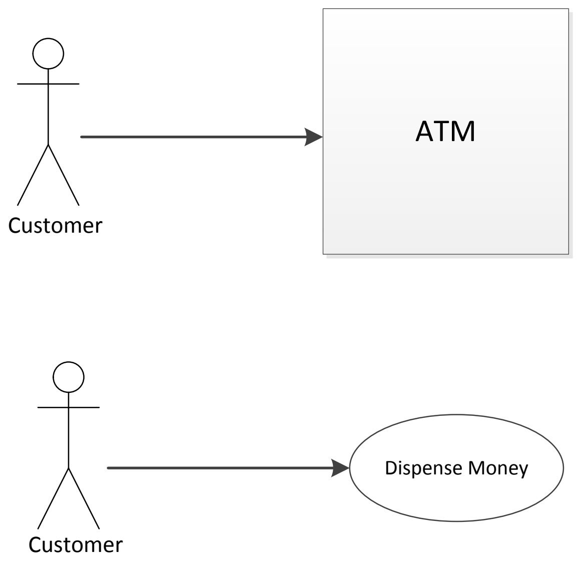
- Please download the project and copy into your Apache “htdocs” folder.
- Find the atm_omid.sql under the root folder for the database file and import into your MySql server.
- Find the database configuration file under “/application/config/database.php” to set the database connection.
- After setting the database connection, open your browser and type “localhost/atm” to run the application.

Design Pattern: MVC
Frame Work: CodeIgniter.

CodeIgniter is designed based on MVC pattern. It is powerful, reliable, secure and provides rich set of libraries.
By following the standard MVC pattern when implementing the project it is easy to create portable components and increase the maintainability of the product.

In the next pages the proper software engineering approach to develop the project were practiced. These consist of the following:

- Requirements are extracted from client document
- Use case diagram is designed
- Basic flow is written based on the requirements and use case
- Sequence diagram is designed from basic flows.
- Class diagram is derived from sequence diagram and were designed based on Model, View, Controller pattern to fit into the framework.
- Implementation of code and testing started together until the product is ready and the test cases covered all possible inputs.



External Interface and Use case Diagram

Basic Flow

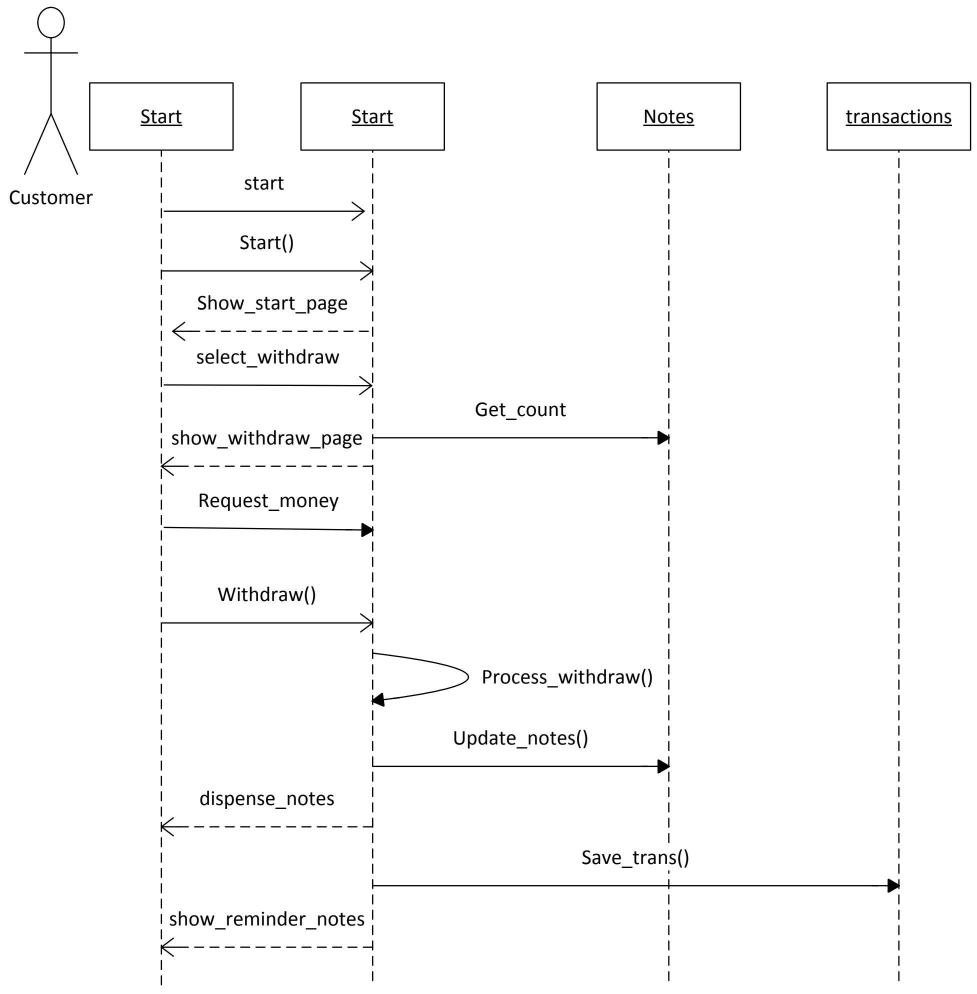
1. This use case will start when the customer inserts the card.
2. The system shows the option to withdraw money. **(REQ_101)**.
3. The system shows the types and count of each type of notes. **(REQ_102)**
4. The customer selects to withdraw money (**REQ_103)**, and the system shows the panel enter the desired amount.
5. The customer enters the desired amount and the system processes the request.
6. The system dispenses the money in combination of $20 and $50 (**REQ_104).**
7. The system updates the available count of each type of note (**REQ_105),** reports it (**REQ_102)** and shows how many of each type of note dispensed (**REQ_106)**. [A1: Invalid Amount] (**REQ_107).**
8. Use case ends.

Alternative Flow

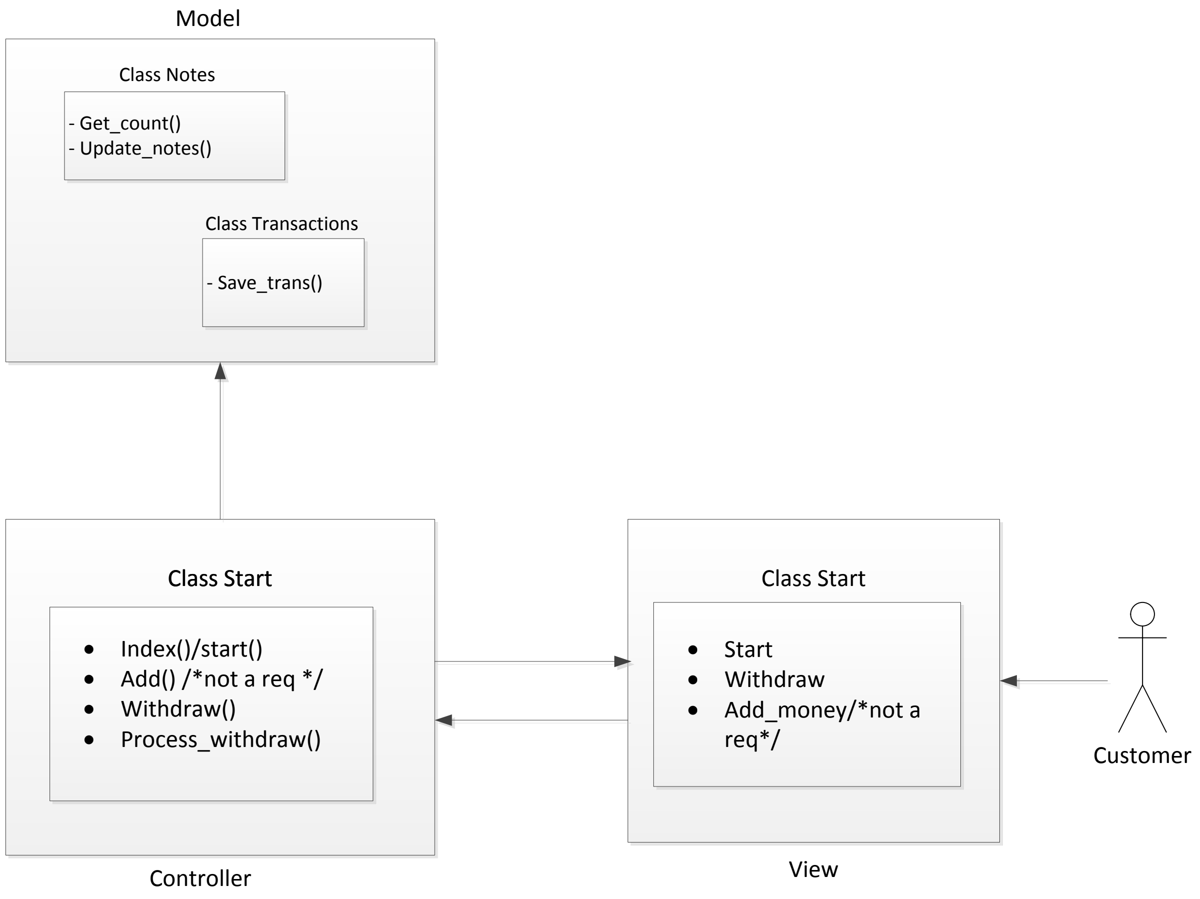
A1: Invalid Amount

1. The system does not dispense money and shows error message (**REQ_107)**.
2. Use case continues.

Overview of the MVC design:



Sequence Diagram



Class Diagram

Testing.

For the availability of 5 x $50 and 5 x $20, test the withdrawal function for the combination of zero count, minimum count, maximum count and some random middle count of $20 and $50 has been performed. Automated test performed by giving the following amount as input.

TEST ACTION	EXPECTED RESULT	STATUS	ACTUAL RESULT
Requests for $0	Show enter an amount	Pass	Same as expected
Requests for $10	Show enter higher amount	Pass	Same as expected
Requests for $20	Show Collect 1 x $20	Pass	Same as expected
Requests for $30	Amount not available	Pass	Same as expected
Requests for $40	Collect 2 x $20	Pass	Same as expected
Requests for $50	Collect 1 x $50	Pass	Same as expected
Requests for $60	Collect 3 x $20	Pass	Same as expected
Requests for $70	Collect 1x$50 and 1x$20	Pass	Same as expected
Requests for $80	Collect 4x$20	Pass	Same as expected
Requests for $90	Collect 1x$50 and 2x$20	Pass	Same as expected
Requests for $100	- Collect 2x$50 - Collect 4x$20	Pass	Same as expected
Requests for $110	Collect 1x$50 and 3x$20	Pass	Same as expected
Requests for $120	Collect 2x$50 and 1x$20	failed/fixed	Collect 6x$20
Requests for $130	Collect 1x$50 and 4x$20	Pass	Same as expected
Requests for $140	Collect 2x$50 and 2x$20	Pass	Same as expected
Requests for $150	- Collect 3x$50 - Collect 1x$50 and 5X$20	Pass	Same as expected
Requests for $160	- Collect 2x$50 and 3x$20 - Collect 1x$50 and 5X$20	Pass	Same as expected
Requests for $170	- Collect 3x$50 and 1x$20	Failed/fixed	Collect 1x$50 and 6x$20
Requests for $180	- Collect 2x$50 and 4x$20 - Collect 2x$50 and 4X$20	Pass	Same as expected
Requests for $240	- Collect 4x$50 and 2x$20	Pass	Same as expected
Requests for $250	- Collect 5x$50 - Collect 3x$50 and 5X$20	Pass	Same as expected
Requests for $270	- Collect 5x$50 and 1x$20	Failed/fixed	Collect 3x$50 and 6X$20
Requests for $280	- Collect 4x$50 and 4X$20	Pass	Same as expected
Requests for $290	- Collect 5x$50 and 2x$20	Pass	Same as expected
Requests for $300	- Collect 4x$50 and 5x$20	Pass	Same as expected
Requests for $310	- Collect 5x$50 and 3x$20	Pass	Same as expected
Requests for $320	- Amount not available	Pass	Same as expected
Requests for $330	- Collect 5x$50 and 4x$20	Pass	Same as expected
Requests for $370	- Amount too high	Pass	Same as expected