

IST 303 Team Project: Part D

HODL (Hold On, Don't Liquidate) App

Team Members:

Eman Alzahrani, Marcus Dashoff,
Raymond Derrick Dimla,
James Hah, Chen Zhu



Project Overview:



Developing a **simple stock trading app** that allows users to:

✓ Buy and sell
stocks


✓ View their
portfolio

✓ Cancel
unfulfilled orders


✓ Analyze
market trends



Key Focus:

Real-time stock price
updates 

Market trends and news


Order management
(buy, sell, cancel) 

Ending an Iteration (Pilone, D., & Miles, R. (2008))



Reviewing Design Patterns

Continue using Model View Controller (MVC)



Exploring new technologies

Pytest

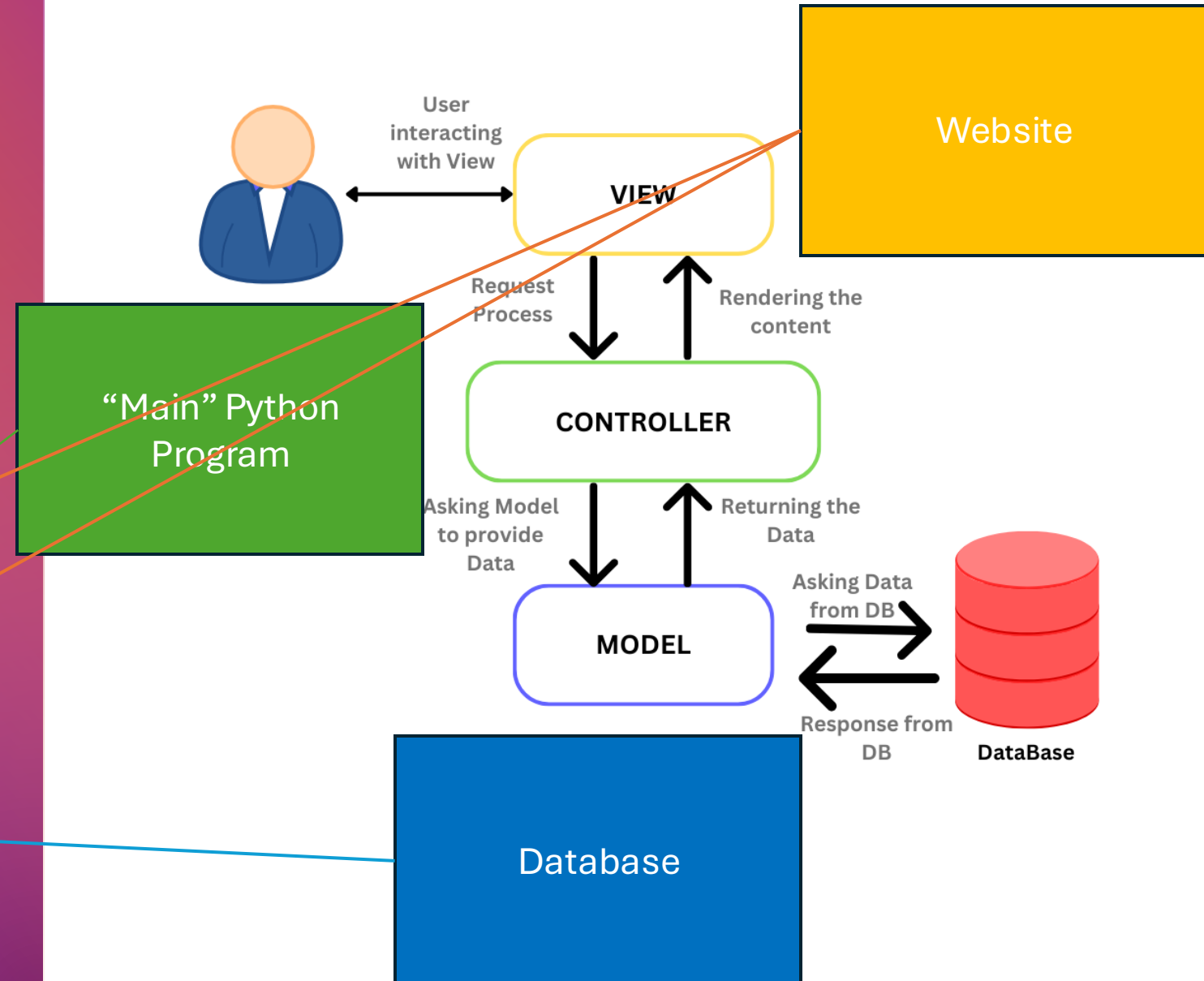


Bugs / Refactoring

Reviewed Design Pattern

- Design pattern
 - Model View Controller (MVC)
- Database design
 - User
 - Stock
 - Purchase
 - Sales
 - Currency
 - User Stock
- Front-end website with basic navigation bar
- User Log In & Session Setting – Flask Login Manager

```
> controllers
> helper
> models
> static
> templates
> tests
> .gitignore
> database.db
> init_db.py
> main.py
> README.md
> schema.sql
```



- **What We Achieved in Iteration 2:**



WIP: Iteration 2.0

- Order Fulfillment
 - Sell orders
- Order History
- Current position in stock
- Cumulative realized P/L for that stock



New Velocity



Forecasted Velocity = .4

14 Days completed

20 Calendar days in iteration

5 Members



Actual Velocity

$$14 / (20 \times 5) = .14$$



$5 \times 40 \times .14 = 28$ days needed for next iteration

Velocity + and Burn Down Chart

Burndown Chart

Team of 5

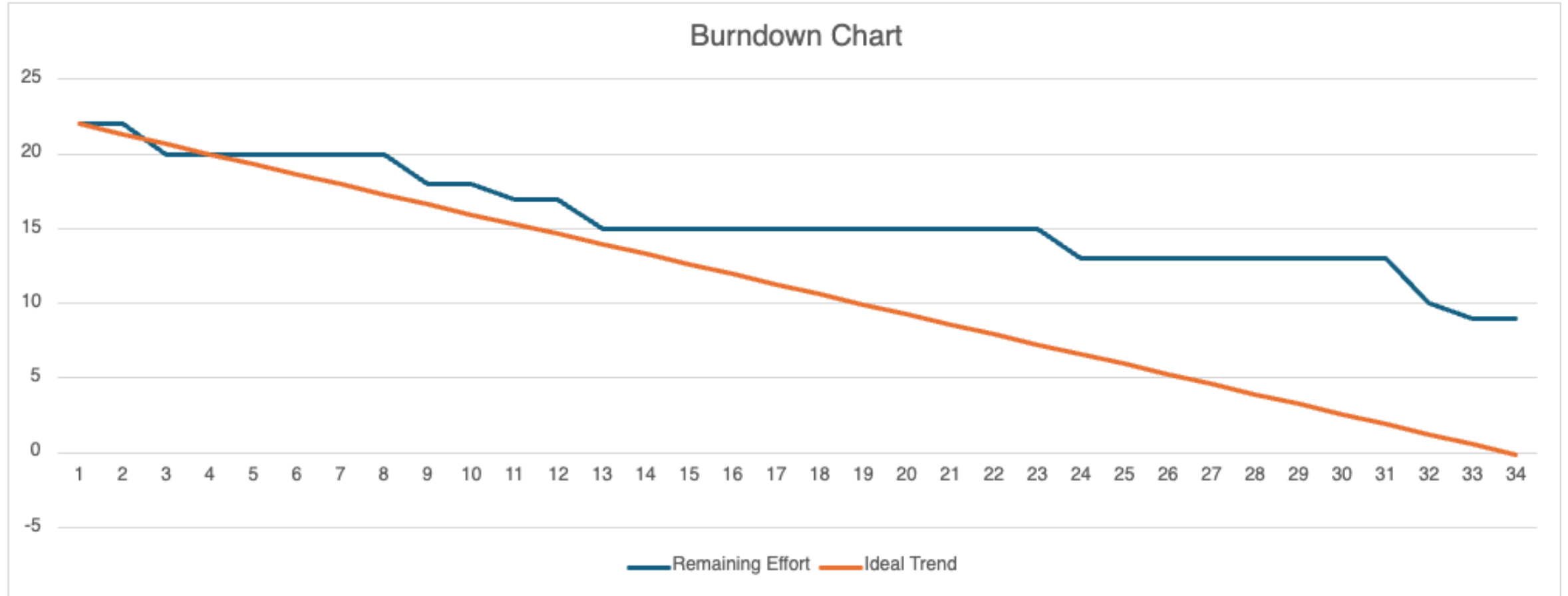
23 Days: 2-3 days per week * 4 weeks

.14 Velocity due to full-time students or fully employed + part-time students

$5 * 40 * .14 = 28$ Working days in our iteration

2 Iteration to milestone 1

Velocity and Burn Down Chart



Stand Up Meetings



3/26/25
LEARNING PYTEST



3/30/25
STORIES + PLANNING
POKER



4/9/25
STORY 4 REVIEW



4/13/25
STORY 5 REVIEW



4/20/25
STORY 6 REVIEW



4/22/25
PRESENTATION
REVIEW




4/23/25
PRESENTATION

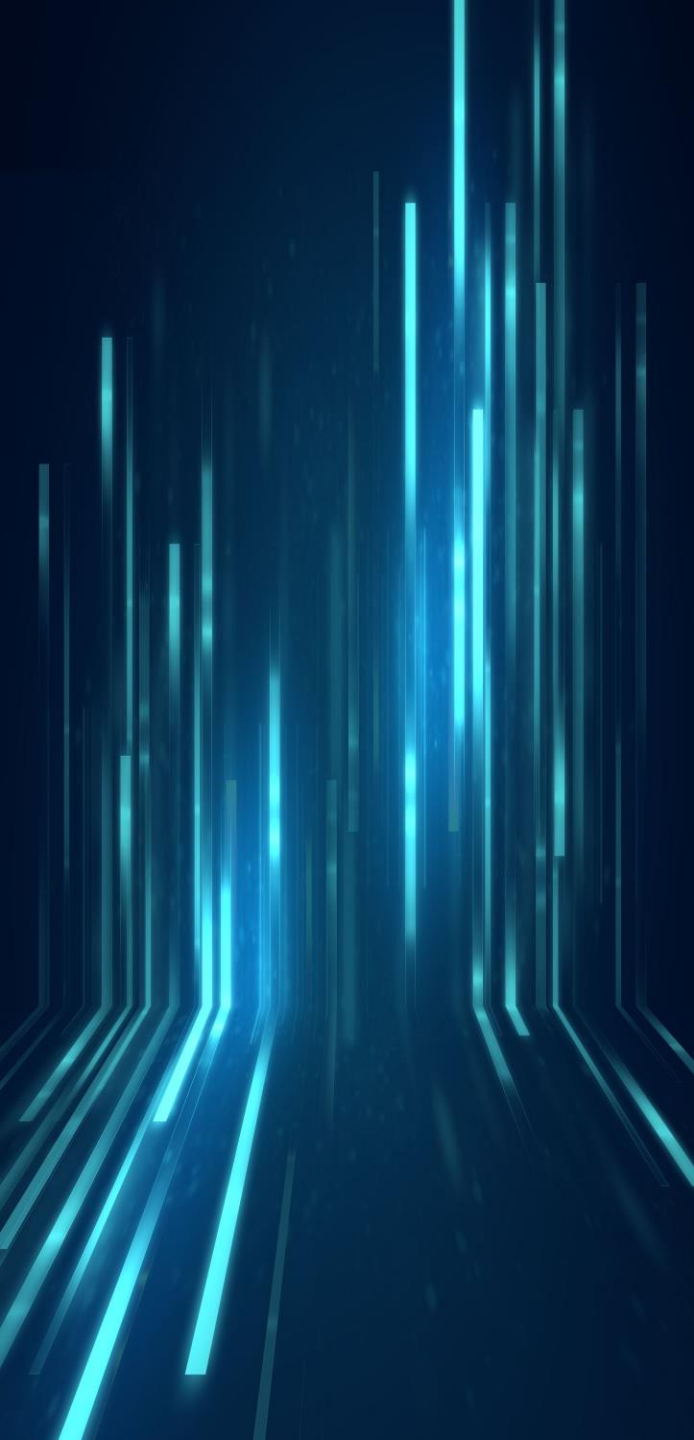


User Story 4

Sell Order

I want to be able to sell my stocks by specifying the stock symbol, the number of shares, and the price at which I want to sell them. After clicking sell, the app creates sell records and saves them while the system processes my order. If I change my mind, I want to be able to see a list of my unfulfilled sell orders, and to cancel any of them.





Story 4 Tasks – **Sell Order**

Tasks:

- Design UI for entering stock symbol, shares, and price.
- Implement functionality to submit sell order.
- Create database schema for sell orders.
- Implement database insertion for sell orders.
- Add functionality to retrieve unfulfilled sell orders from the database.
- Implement functionality to cancel sell orders.
- Update database to reflect canceled orders.
- Display unfulfilled sell orders in the UI.
- Write unit tests for sell order submission and order cancellation.

Software Demonstration

- What the code does
- How it fulfills the user stories






User Story 5

Order Fulfillment Logic

When I create a purchase or sell order, I want the app to search for and fulfill the order, and to automatically update my stock holdings, so that the transaction is executed properly and my portfolio and my portfolio reflects the accurate changes.





Story 5 Tasks – **Order Fulfillment Logic**

Tasks:

- Implement order search functionality.
- Implement order fulfillment logic.
- Create/update database schema for user stock holdings.
- Implement functionality to update user stock holdings in the database.
- Write unit tests for order fulfillment.
- Write unit tests for updating user stock holdings.

Software Demonstration

- What the code does
- How it fulfills the user stories






User Story 6

Order History and Portfolio Performance

I want to see a list of my past fulfilled transactions, my current account balance, and a display of my current stock holdings, so that I can review my trading history, monitor my funds, and understand my investment portfolio.





Story 6 Tasks – Order History and Portfolio Performance

Tasks:

- Implement functionality to retrieve past fulfilled transactions from the database.
- Display past fulfilled transactions in the UI.
- Implement functionality to retrieve user's current balance.
- Display user's current balance in the UI.
- Implement functionality to retrieve user's current stock holdings from the database.
- Display user's current stock holdings in the UI.
- Write unit tests for retrieving and displaying order history, balance, and holdings.

Software Demonstration

- What the code does
- How it fulfills the user stories



Tests

Pytest Highlights

- **Total 94% test coverage** on all routes in main.py
- (White) Each feature has its own test file: purchase, sell, search, fulfillment, main
- (Grey) Simulated login using session mocking
- (White) DB reset before test to avoid flaky tests
- One test file per controller
- Handled edge cases: wrong password, no balance, missing stocks
- (White) Verified order matching logic with DB state checks

Team Agile Retro



Principle 12

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Success

- Great Agile approach with good project management
- Weekly Team Sync Up
- Communication between Product and Devs

Failure

- DO NOT OVER PROMISE
- Devs need to learn how to push back

Other takeaways

- It is okay to spend time spiking and investigating before coding
- **Do not directly push to the main branch**
 - Take advantage of Pull Request with merge

A message from our Client



Posted Apr 2, 3:28pm

Group C, we think this app can be what we need to save the company. Thank you for demoing the project. Our team has discussed what we would like out of the app, and it is important that when we search for a ticker we are presented with our current position in the stock (if any) as well as our cumulative realized P/L for that stock. We have had too many cases where individuals are repeatedly getting let down by the same stocks. It is not their fault, we cannot be expected to remember this.

Sincerely,

your amazing client

Response:

- Due to the database holding static data, there is no historical data that can be used to show position + P/L.
- This will be worked on during our next iteration, once internet/API functions have been worked on.



Questions

Conclusion