LAB

ANCHOR AND TOKENS

1. Make a new anchor project, and run the default tests.

```
anchor init escrow --template=multiple

cd escrow

anchor test
```

2. Add Anchor SPL, to use SPL tokens!

Open programs/escrow/Cargo.toml

```toml
[dependencies]
anchor-lang = { version = "0.30.0", features = ["init-if-needed"]}
anchor-spl = "0.30.0"
```

And also add this to the [features] sections

```toml
idl-build = ["anchor-lang/idl-build", "anchor-spl/idl-build"]
```

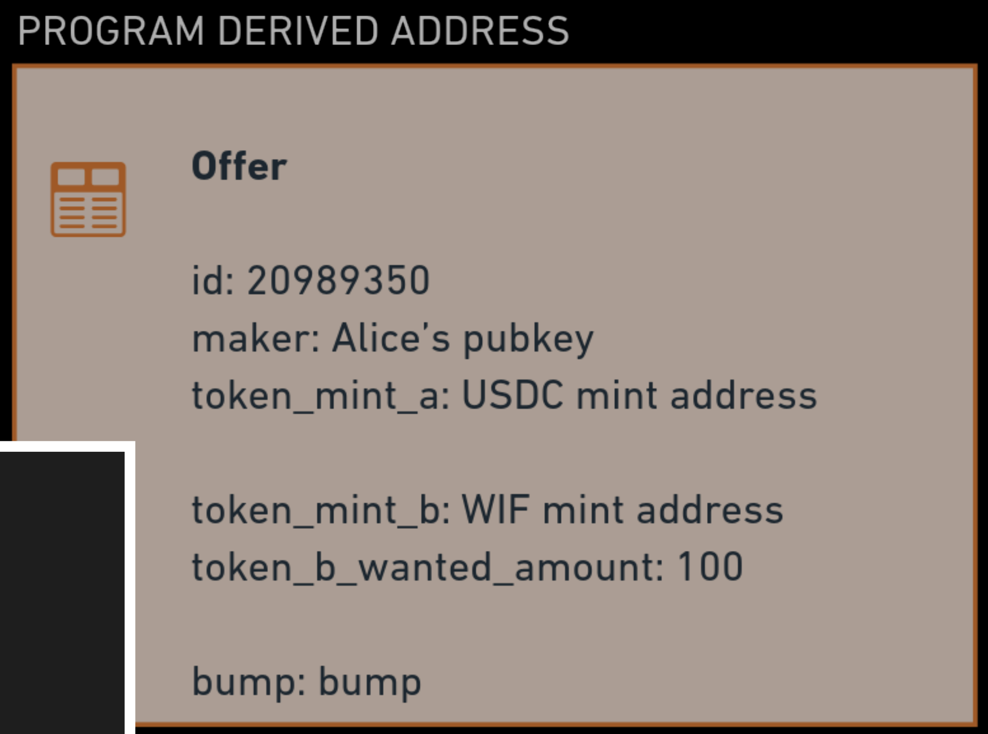
Open programs/escrow/src/constants.rs and add

```rust
pub const ANCHOR_DISCRIMINATOR: usize = 8;
```

3. Create the `Offer` struct

Make programs/escrow/src/state/offer.rs

```rust
use anchor_lang::prelude::*;

#[account]
#[derive(InitSpace)]
pub struct Offer {
    pub id: u64,
    pub maker: Pubkey,
    pub token_mint_a: Pubkey,
    pub token_mint_b: Pubkey,
    pub token_b_wanted_amount: u64,
    pub bump: u8,
}
```


PROGRAM DERIVED ADDRESS

Offer

id: 20989350
maker: Alice's pubkey
token_mint_a: USDC mint address

token_mint_b: WIF mint address
token_b_wanted_amount: 100

bump: bump

Also update programs/escrow/src/state/mod.rs

```rust
pub mod offer;
pub use offer::*;
```

4. Create our make_offer file!

Edit programs/escrow/src/instructions/mod.rs

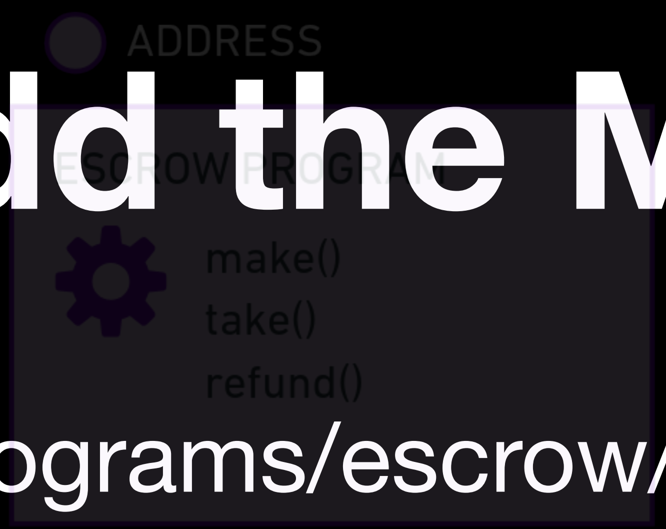
```rust
pub mod make_offer;
pub use make_offer::*;
```

Create programs/escrow/src/instructions/make_offer.rs

```rust
use anchor_lang::prelude::*;

use anchor_spl::{
    associated_token::AssociatedToken,
    token_interface::{transfer_checked, Mint, TokenAccount, TokenInterface,
TransferChecked},
};

use crate::{Offer, ANCHOR_DISCRIMINATOR};
```

continued on next slide...

5. Add the MakeOffer accounts struct 1/2

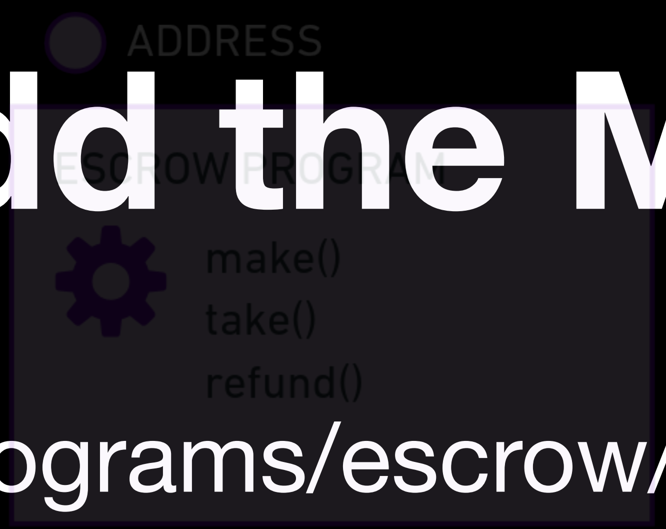


programs/escrow/src/instructions/make_offer.rs

```rust
#[derive(Accounts)]
#[instruction(id: u64)]
pub struct MakeOffer<'info> {
    #[account(mut)]
    pub maker: Signer<'info>,

    #[account(mint::token_program = token_program)]
    pub token_mint_a: InterfaceAccount<'info, Mint>,

    #[account(mint::token_program = token_program)]
    pub token_mint_b: InterfaceAccount<'info, Mint>,

    #[account(
        mut,
        associated_token::mint = token_mint_a,
        associated_token::authority = maker,
        associated_token::token_program = token_program
    )]
    pub maker_token_account_a: InterfaceAccount<'info, TokenAccount>,
```

continued on next slide...

6. Add the MakeOffer accounts struct 2/2



programs/escrow/src/instructions/make_offer.rs continued (inside MakeOffer struct)

```
    #[account(
        init,
        payer = maker,
        space = ANCHOR_DISCRIMINATOR + Offer::INIT_SPACE,
        seeds = [b"offer", maker.key().as_ref(), id.to_le_bytes().as_ref()],
        bump
    )]
    pub offer: Account<'info, Offer>,

    #[account(
        init,
        payer = maker,
        associated_token::mint = token_mint_a,
        associated_token::authority = offer,
        associated_token::token_program = token_program
    )]
    pub vault: InterfaceAccount<'info, TokenAccount>,

    pub associated_token_program: Program<'info, AssociatedToken>,
    pub token_program: Interface<'info, TokenInterface>,
    pub system_program: Program<'info, System>,
}
```

continued on next slide...

7. Create send_offered_tokens_to_vault()!

programs/escrow/src/instructions/make_offer.rs

```rust
pub fn send_offered_tokens_to_vault(
    context: &Context<MakeOffer>,
    token_a_offered_amount: u64,
) -> Result<()> {
    let transfer_accounts = TransferChecked {
        from: context.accounts.maker_token_account_a.to_account_info(),
        mint: context.accounts.token_mint_a.to_account_info(),
        to: context.accounts.vault.to_account_info(),
        authority: context.accounts.maker.to_account_info(),
    };

    let cpi_context = CpiContext::new(
        context.accounts.token_program.to_account_info(),
        transfer_accounts,
    );

    transfer_checked(
        cpi_context,
        token_a_offered_amount,
        context.accounts.token_mint_a.decimals,
    )
}
```

continued on next slide...

8. Create save_offer()!

programs/escrow/src/instructions/make_offer.rs

```rust
pub fn save_offer(context: Context<MakeOffer>, id: u64, token_b_wanted_amount: u64) ->
Result<()> {
    context.accounts.offer.set_inner(Offer {
        id,
        maker: context.accounts.maker.key(),
        token_mint_a: context.accounts.token_mint_a.key(),
        token_mint_b: context.accounts.token_mint_b.key(),
        token_b_wanted_amount,
        bump: context.bumps.offer,
    });
    Ok(())
}
```

9. Update our lib.rs

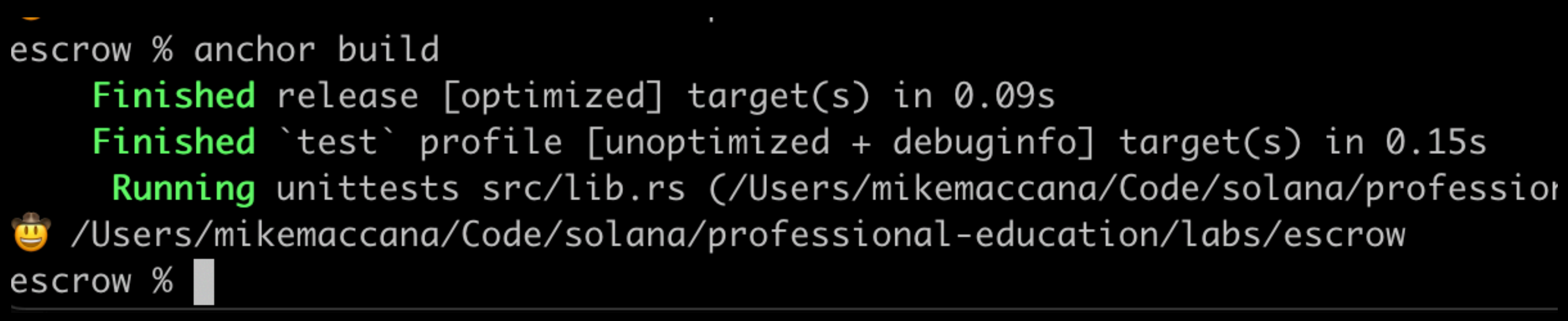
Replace the entire 'mod' in programs/escrow/src/lib.rs:

```rust
#[program]
pub mod escrow {
    use super::*;

    pub fn make_offer(
        context: Context<MakeOffer>,
        id: u64,
        token_a_offered_amount: u64,
        token_b_wanted_amount: u64,
    ) -> Result<()> {
        instructions::make_offer::send_offered_tokens_to_vault(&context,
token_a_offered_amount)?;
        instructions::make_offer::save_offer(context, id, token_b_wanted_amount)
    }
}
```

continued on next slide...

10. Check this compiles!

```
anchor build
```



```
escrow % anchor build
    Finished release [optimized] target(s) in 0.09s
    Finished `test` profile [unoptimized + debuginfo] target(s) in 0.15s
     Running unittests src/lib.rs (/Users/mikemaccana/Code/solana/professio
🤠 /Users/mikemaccana/Code/solana/professional-education/labs/escrow
escrow % 
```

11. Add some tests!

The test files involves a lot of set up

Grab tests from
https://github.com/solana-developers/professional-education/blob/main/labs/escrow/tests/escrow.ts

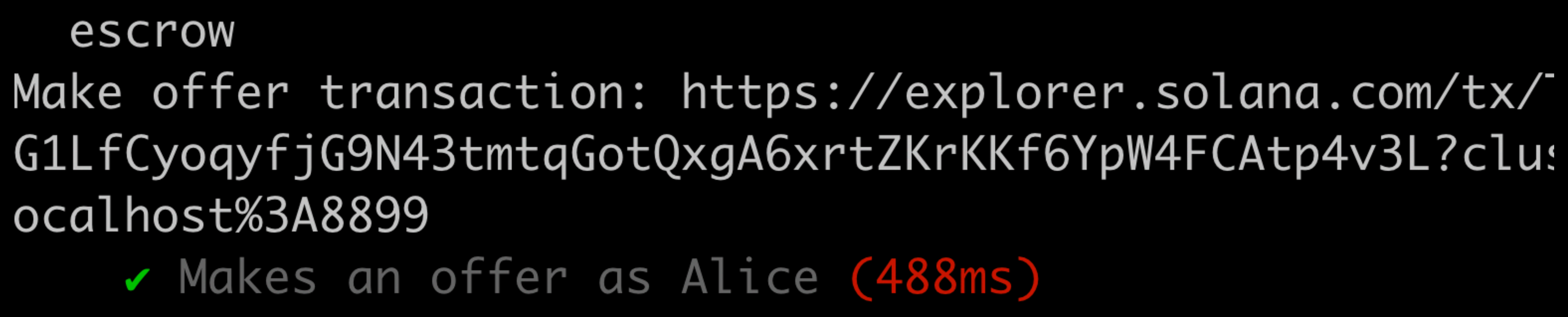
Save to tests/escrow.ts

Comment out the take tests for now!

12. Run the tests with

```
anchor test
```

Ensure your tests pass!



```
  escrow
Make offer transaction: https://explorer.solana.com/tx/
G1LfCyoqyfjG9N43tmtqGotQxgA6xrtZKrKKf6YpW4FCAtp4v3L?clu
ocalhost%3A8899
    ✔ Makes an offer as Alice (488ms)
```

13. Create our take_offer file!

Edit programs/escrow/src/instructions/mod.rs and add:

```
pub mod take_offer;
pub use take_offer::*;
```

Create programs/escrow/src/instructions/take_offer.rs

```
use anchor_lang::prelude::*;

use anchor_spl::{
    associated_token::AssociatedToken,
    token_interface::{
        close_account, transfer_checked, CloseAccount, Mint, TokenAccount,
TokenInterface,
        TransferChecked,
    },
};

use crate::Offer;
```

continued on next slide...

14. Create our TakeOffer accounts struct 1/3

programs/escrow/src/instructions/take_offer.rs continued

```rust
#[derive(Accounts)]
pub struct TakeOffer<'info> {
    #[account(mut)]
    pub taker: Signer<'info>,

    #[account(mut)]
    pub maker: SystemAccount<'info>,

    pub token_mint_a: InterfaceAccount<'info, Mint>,

    pub token_mint_b: InterfaceAccount<'info, Mint>,

    #[account(
        init_if_needed,
        payer = taker,
        associated_token::mint = token_mint_a,
        associated_token::authority = taker,
        associated_token::token_program = token_program,
    )]
    pub taker_token_account_a: Box<InterfaceAccount<'info, TokenAccount>>,

    #[account(
        mut,
        associated_token::mint = token_mint_b,
        associated_token::authority = taker,
        associated_token::token_program = token_program,
    )]
    pub taker_token_account_b: Box<InterfaceAccount<'info, TokenAccount>>,
```

continued on next slide...

15. Create our TakeOffer accounts struct 2/3

programs/escrow/src/instructions/take_offer.rs continued (inside TakeOffer struct)

```rust
    #[account(
        init_if_needed,
        payer = taker,
        associated_token::mint = token_mint_b,
        associated_token::authority = maker,
        associated_token::token_program = token_program,
    )]
    pub maker_token_account_b: Box<InterfaceAccount<'info, TokenAccount>>,

    #[account(
        mut,
        close = maker,
        has_one = maker,
        has_one = token_mint_a,
        has_one = token_mint_b,
        seeds = [b"offer", maker.key().as_ref(), offer.id.to_le_bytes().as_ref()],
        bump = offer.bump
    )]
    offer: Account<'info, Offer>,
```

continued on next slide...

16. Create our TakeOffer accounts struct 3/3

programs/escrow/src/instructions/take_offer.rs continued (inside TakeOffer)

```rust
    [account(
        mut,
        associated_token::mint = token_mint_a,
        associated_token::authority = offer,
        associated_token::token_program = token_program,
    )]
    pub vault: InterfaceAccount<'info, TokenAccount>,

    pub associated_token_program: Program<'info, AssociatedToken>,
    pub token_program: Interface<'info, TokenInterface>,
    pub system_program: Program<'info, System>,
}
```

All finished!

17. Create send_wanted_tokens_to_maker()

programs/escrow/src/instructions/take_offer.rs continued

```
pub fn send_wanted_tokens_to_maker(ctx: &Context<TakeOffer>) -> Result<()> {
    let transfer_accounts = TransferChecked {
        from: ctx.accounts.taker_token_account_b.to_account_info(),
        mint: ctx.accounts.token_mint_b.to_account_info(),
        to: ctx.accounts.maker_token_account_b.to_account_info(),
        authority: ctx.accounts.taker.to_account_info(),
    };

    let cpi_ctx = CpiContext::new(
        ctx.accounts.token_program.to_account_info(),
        transfer_accounts,
    );

    transfer_checked(
        cpi_ctx,
        ctx.accounts.offer.token_b_wanted_amount,
        ctx.accounts.token_mint_b.decimals,
    )
}
```

continued on next slide...

18. Create withdraw_and_close_vault() 1/2

programs/escrow/src/instructions/take_offer.rs continued

```rust
pub fn withdraw_and_close_vault(ctx: Context<TakeOffer>) -> Result<()> {
    let signer_seeds: [&[&[u8]]; 1] = [&[
        b"offer",
        ctx.accounts.maker.to_account_info().key.as_ref(),
        &ctx.accounts.offer.id.to_le_bytes()[..],
        &[ctx.accounts.offer.bump],
    ]];

    let accounts = TransferChecked {
        from: ctx.accounts.vault.to_account_info(),
        mint: ctx.accounts.token_mint_a.to_account_info(),
        to: ctx.accounts.taker_token_account_a.to_account_info(),
        authority: ctx.accounts.offer.to_account_info(),
    };

    let cpi_context = CpiContext::new_with_signer(
        ctx.accounts.token_program.to_account_info(),
        accounts,
        &signer_seeds,
    );
```

continued on next slide...

19. Create withdraw_and_close_vault() 2/2

programs/escrow/src/instructions/take_offer.rs continued (inside withdraw_and_close_vault)

```rust
transfer_checked(
        cpi_context,
        ctx.accounts.vault.amount,
        ctx.accounts.token_mint_a.decimals,
    )?;

    let accounts = CloseAccount {
        account: ctx.accounts.vault.to_account_info(),
        destination: ctx.accounts.taker.to_account_info(),
        authority: ctx.accounts.offer.to_account_info(),
    };

    let cpi_context = CpiContext::new_with_signer(
        ctx.accounts.token_program.to_account_info(),
        accounts,
        &signer_seeds,
    );

    close_account(cpi_context)
}
```

20. Update lib.rs to add take_offer()

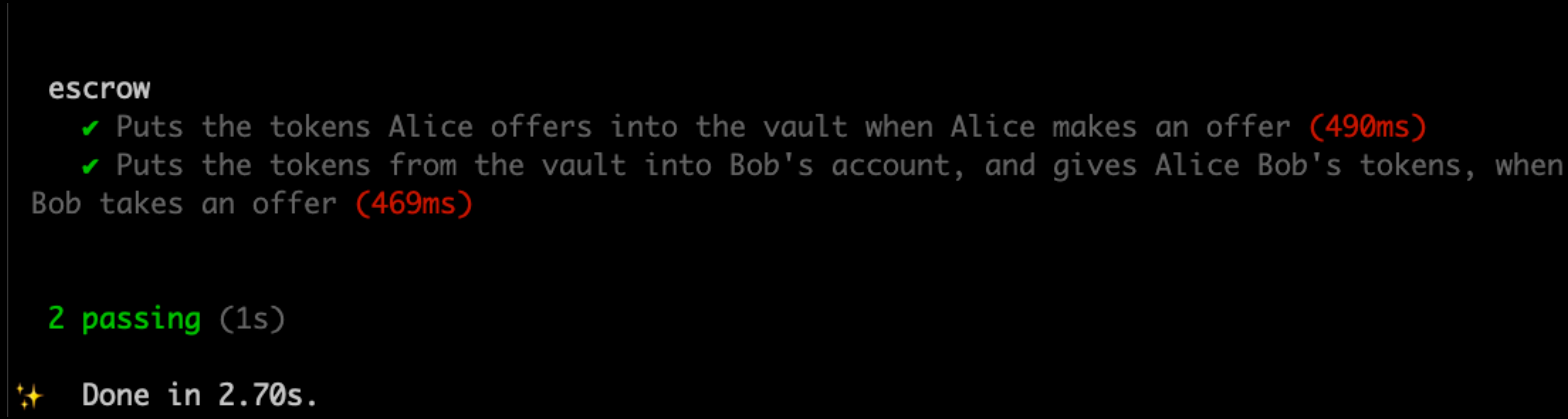
Now programs/escrow/src/lib.rs (inside the 'mod escrow')

```rust
pub fn take_offer(context: Context<TakeOffer>) -> Result<()> {
    instructions::take_offer::send_wanted_tokens_to_maker(&context)?;
    instructions::take_offer::withdraw_and_close_vault(context)
}
```

Uncommment the 'take' tests and then...

21. Run the tests again!

```
anchor test
```



```
  escrow
    ✔ Puts the tokens Alice offers into the vault when Alice makes an offer (490ms)
    ✔ Puts the tokens from the vault into Bob's account, and gives Alice Bob's tokens, when
 Bob takes an offer (469ms)


  2 passing (1s)

✨  Done in 2.70s.
```